The Dissertation Committee for Peter Bond Backlund Certifies that this is the approved version of the following dissertation:

A Classifier-Guided Sampling Method for Early-Stage Design of Shipboard Energy Systems

Committee:

Carolyn C. Seepersad, Supervisor

Thomas M. Kiehne

Matthew I. Campbell

Richard H. Crawford

John R. Howell

A Classifier-Guided Sampling Method for Early-Stage Design of Shipboard Energy Systems

by

Peter Bond Backlund, B.S.M.E.; M.S.E.

Dissertation

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

Doctor of Philosophy

The University of Texas at Austin

December 2012

Dedication

To my grandfather James Bennet Backlund

Acknowledgements

I would like to thank my advisor, Dr. Seepersad, for all of her guidance, support, and feedback that have made this achievement possible. Dr. Seepersad's dedication to her students in the lab and in the classroom is truly extraordinary. I would also like to thank Dr. Kiehne for his expertise and help with the all-electric ship project. The time and insights provided by my committee members are also greatly appreciated. David Shahan spent many hours with me discussing research ideas, and I am grateful for his time and willingness to share his deep understanding of complex subjects. Additional gratitude is given to Michael Pierce who was extremely helpful when I was learning to use the DTMS framework. I would also like to thank Bill Shutt and Applied Research Laboratories for supporting my travel to several conferences and workshops. The Office of Naval Research is also acknowledged for supporting the Electric Ship Research and Development Consortium. I am thankful for the support of The University of Texas at Austin and the Cockrell School of Engineering for the additional funding I received through the Engineering Doctoral Fellowship program. I would also like to thank Julia O'Rourke, Cassandra Telenko, Andy Tilstra, Nate Putnam, Moss Shimek, Ben Gully, and all of my other labmates for being great friends and colleagues. Lastly, I would like to thank my parents and my brother for their support and encouragement. This achievement would not have been possible without them.

A Classifier-Guided Sampling Method for Early-Stage Design of Shipboard Energy Systems

Peter Bond Backlund, Ph.D.

The University of Texas at Austin, 2012

Supervisor: Carolyn C. Seepersad

The United States Navy is committed to developing technology for an All-Electric Ship (AES) that promises to improve the affordability and capability of its next-generation warships. With the addition of power-intensive 21st century electrical systems, future thermal loads are projected to exceed current heat removal capacity. Furthermore, rising fuel costs necessitate a careful approach to total-ship energy management. Accordingly, the aim of this research is to develop computer tools for early-stage design of shipboard energy distribution systems. A system-level model is developed that enables ship designers to assess the effects of thermal and electrical system configurations on fuel efficiency and survivability. System-level optimization and design exploration, based on these energy system models, is challenging because the models are sometimes computationally expensive and characterized by discrete design variables and discontinuous responses. To address this challenge, a classifier-guided sampling (CGS) method is developed that uses a Bayesian classifier to pursue solutions with desirable performance characteristics. The CGS method is tested on a set of example problems and applied to the AES energy system model. Results show that the CGS method significantly improves the rate of convergence towards known global optima, on average, when compared to genetic algorithms.

Table of Contents

List of Tables

List of Figures

Chapter 1: Introduction

The task of complex engineering system design poses an array of significant challenges to designers. Subsystem interactions, multiple design objectives, uncertain operational environments, computational expense, and high-dimensional design spaces are just a few of the challenges one can expect to encounter when approaching a complex system design problem. In this dissertation, methods are studied and developed to address a few of these challenges. In particular, high-dimensional metamodeling techniques are studied as a means to provide fast approximations of high-dimensional, computationally expensive computer models. However, traditional metamodels are not readily applicable to problems with discrete variables and discontinuous objective functions, and a novel classifier-guided sampling method is developed for optimization and design space exploration of these types of problems. A notional United States Navy All-Electric Ship (AES) is selected as a platform for which metamodels and the classifier-guided sampling method are applied. The AES is ideal for the research presented herein because it is a relatively new technology that features many of the typical characteristics that are inherent to complex engineering system design. Namely, it features a variety of interacting subsystems, discrete system-level design variables, and computationally expensive subsystem and component models.

1.1 All-Electric Ship System-Level Energy Management

The United States Navy is committed to developing technology for an All-Electric Ship (AES) that will eventually replace the current fleet of conventional warships. The goal of this research effort is to develop a futuristic fleet of naval surface ships with increased system efficiencies, decreased operating costs, and improved mission capabilities [1].

The primary difference between the AES and a traditional ship is that the AES features the Integrated Power System (IPS) architecture [2]. In traditional "mechanical drive" ships, a large portion of the installed power generation capacity is strictly dedicated to mechanical propulsion. Smaller auxiliary generators provide the necessary electric power for ship service loads. The difference between an IPS and traditional system is that all of the prime movers are coupled to generators capable of producing large amounts of electric power. The additional available electric power can then be used for electric propulsion, advanced power electronics, electromagnetic weaponry, and regular ship service loads. The use of the IPS architecture results in a ship with fewer prime movers, decreased lifecycle costs, and a more survivable and reconfigurable system architecture [3].

The addition of power-intensive electrical systems creates significant new challenges in the area of total-ship energy management. In particular, heat loads on the future AES are estimated to require a 700% increase in cooling capacity [4]. Due to strict volumetric size constraints, thermal management issues can no longer be solved in an *ad hoc* fashion in which chilling plants are added until thermal loads are satisfied. Novel energy management solutions are needed to prevent catastrophic failure of electrical systems and components [5].

The focus of this research is on the energy distribution systems. When considering the energy distribution systems, the three most important distributed commodities are electric power, chilled water, and refrigerated air. These three systems operate hierarchically such that each child system is dependent on its parent to function (Figure 1.1). Figure 1.1 shows that, despite its name, all of the energy on the AES originates from conventional fossil fuel.

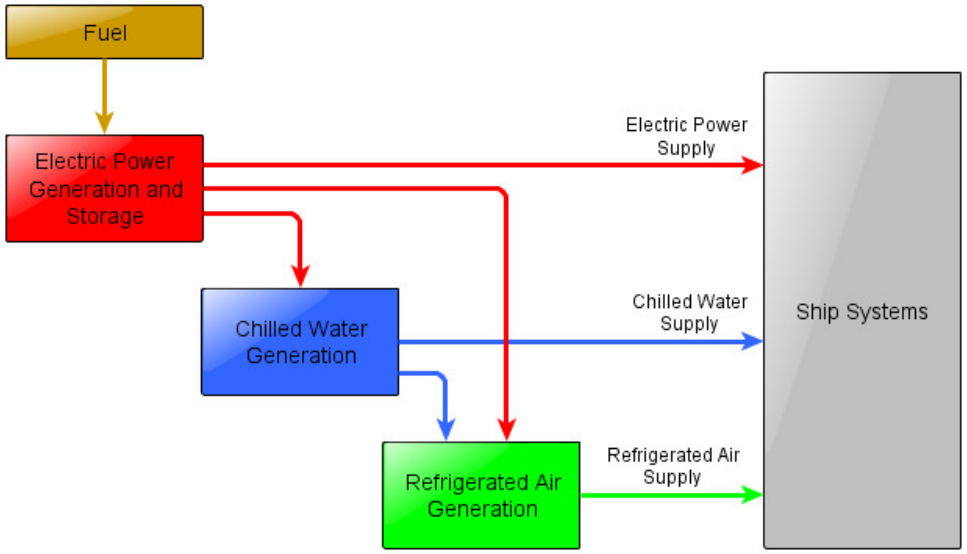


Figure 1.1: Interdependence of distributed energy system layers

It is advantageous to model all three system layers in a single simulation because the thermal and electrical distributed commodities have system-level interdependencies that must be captured by the modeling framework. For example, electric power is required for the generation and distribution of chilled water, and chilled water and electric power are required for the generation of refrigerated air through heat exchangers and distribution fans, respectively. The operating level of one layer may affect demand of another layer. For example, electrical loads may require chilled water to operate at safe operating temperatures. Therefore, chilled water demand may be proportional to electric power demand.

To address the issue of total-ship energy management, tools for assessing ship configurations and connectivity are needed. In this research, configuration refers to the selection and placement of thermal and electrical generation and storage components. The resulting arrangement of an optimally designed AES may differ significantly from a traditional, mechanical drive ship [6]. Once the components are selected, the

connectivity of system component must be altered in real-time during operation to achieve optimal performance in response to different operating modes. Connectivity is determined by the states of automated switches and valves that control the flow of a distributed commodity throughout the system. Configuration of shipboard energy systems is a critical task, and it must be done effectively in order to restore power to areas affected by battle damage, to maintain optimal operating efficiency, and to reroute distributed energy commodities to maximize the capability of the vessel [7].

To gain an understanding of the types of variables that are involved in total-ship energy system design, consider the zonal distribution architecture shown in Figure 1.2. Each energy distribution layer (electrical, chilled water, refrigerated air) has its own zonal distribution architecture, where the location of each load, generation component, and storage component is defined by the zone in which it is located.

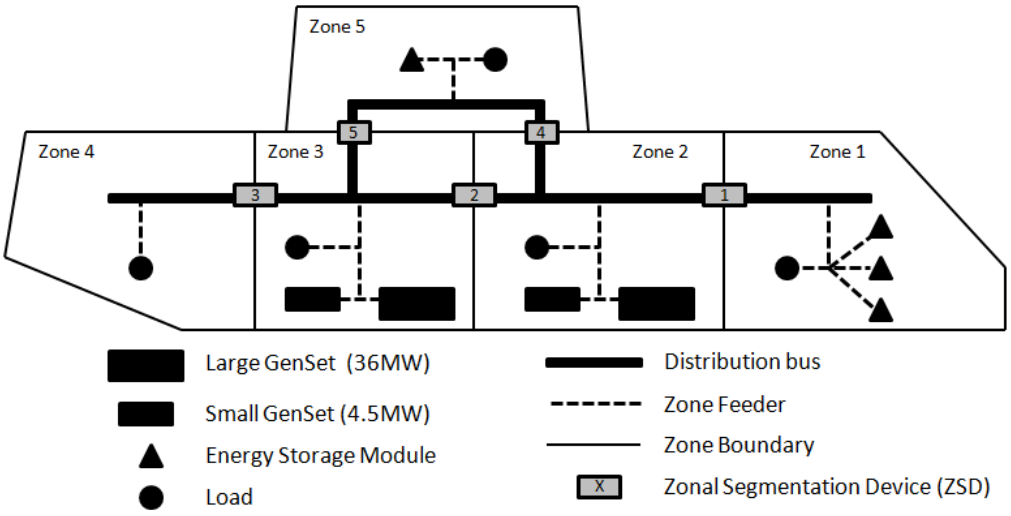


Figure 1.2: Zonal energy distribution architecture

Using the zonal modeling approach, the configuration and connectivity variables are represented as sets of vectors and matrices that can be easily manipulated by optimization or design exploration algorithms. For example, in Figure 1.2, there are two

4.5MW and two 36MW generators. One of each type of generator is placed in Zones 2 and 3. There are three electrical energy storage modules in Zone 1, and one storage module in Zone 5. Therefore, the matrices that represent the configuration in Figure 1.2 are

$$Gens = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \end{bmatrix}$$
$$Storage = \begin{bmatrix} 3 & 0 & 0 & 0 & 1 \end{bmatrix}$$

where *Gens* is a matrix whose first row represents the quantity of 4.5 MW generators and second row represents the quantity of 36 MW generators in each zone, and *Storage* is a vector used to represent the number of storage modules in each zone.

Zonal segmentation devices (ZSDs) are an important feature of the zonal modeling approach. The ZSDs are placed at the zone borders and simulate valves and switches that are either closed or open. The ZSDs play a critical role in the system-level performance studies performed in this research. The ZSDs can be configured to isolate certain zones or segregate entire portions of the ship. A damaged zone can be simulated and isolated by setting that zone's adjacent ZSDs to the closed position. Zone segmentation may improve operating efficiency in certain situations by optimally matching total connected loads with total connected generation capacity. For example, consider a scenario in which it is more efficient for the generators in Zone 2 to serve only Zones 1 and 2 and for the generators in Zone 3 to serve Zones 3, 4, and 5. In this case, the states of the ZSDs could be set to

$$ZSD = \begin{bmatrix} 1 & 0 & 1 & 0 & 1 \end{bmatrix}$$

where *ZSD* is a vector representing the positions of the segmentation switches. In this case, ZSDs 2 and 4 are in the OFF position, meaning that no electric power is able to

flow across these zone boundaries. Note that the values contained in the component matrices are restricted to integer values, and the values in the *ZSD* vector are binary.

The zonal modeling approach is used because the zonal distribution architecture is becoming increasingly common in present-day ships. Studies have shown that zonal electrical distribution architectures result in systems with lower acquisition costs, lower weight, and better operational flexibility [8]. The benefits of a zonal architecture are not limited to electrical distribution systems. For example, Shiffler [9] estimated that a zonal architecture for a shipboard fire main system resulted in cost and weight savings of 20% and 9%, respectively. Zonal electrical distribution architectures have been implemented in several classes of U.S. Navy warships, including DDG-51 guided missile destroyers, LHD-8 assault ships, and LPD-17 transport vessels [10].

While the physical zone boundaries remain the same for each distribution layer, each electrical, chilled water, and refrigerated air system has its own unique zonal layout. Therefore the total number of possible designs and configurations is rather large, thus motivating the need for specialized design tools.

In this research, design is performed at the system level to pursue configurations that have the following desired operating characteristics: survivability and operating efficiency. Effective execution of energy system design is necessary to optimize these objectives.

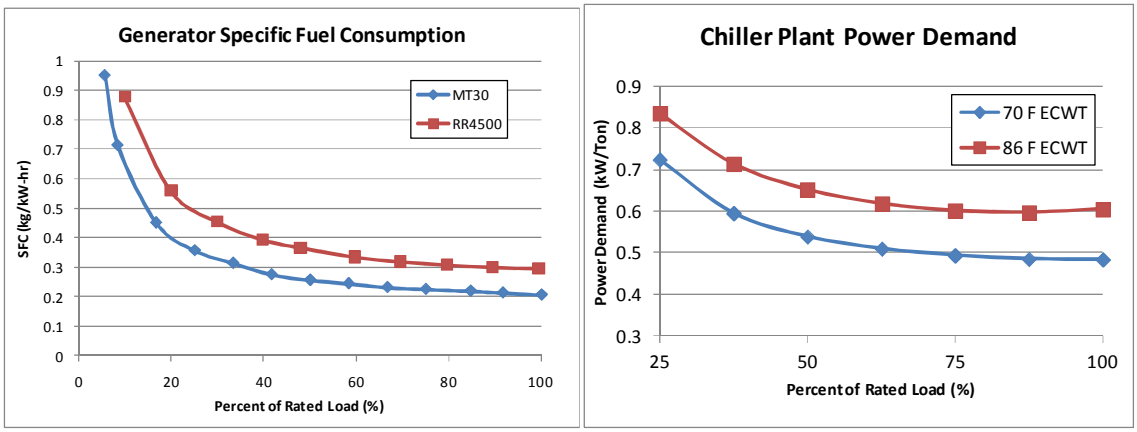
Survivability is a measure of the extent to which a shipboard energy system can achieve a desired outcome in the context of unforeseen damage scenarios. Damage scenarios can include damage to power generation equipment, energy storage components, and energy distribution components. In some cases, it may be impossible to restore electrical or cooling power to all areas of the ship. The lethality of an AES is closely linked to the survivability of the ship energy system; future AES weapon systems,

such as railguns and free-energy lasers, require significant amounts of electric power and thermal cooling capacity to function.

There are various metrics available for assessing survivability. A simple metric is to define a desired outcome to a specific threat and assess survivability based on whether or not the desired outcome is achieved [11]. A more detailed approach is to rank all ship systems based on their mission criticality. In a damage situation, loads are filled in order of priority. Survivability can then be determined by observing the highest priority loads that cannot be filled [12]. For the purposes of the research presented here, survivability shall be determined by observing the total amount of electrical or cooling demand that goes unmet during a notional battle damage scenario.

Operating efficiency refers to the rate of fuel consumption of a shipboard distributed energy system during normal operating conditions. The part-load efficiencies of electric power and chilled water generation components have the most significant effect on operating efficiency. In Figure 1.3, gas turbine generator and chiller plant efficiencies are shown as a function of percent of rated load.

In Figure 1.3a, specific fuel consumption (SFC) curves for the Rolls Royce MT-30 (36 MW) and RR-4500 (4.5 MW) generators are shown. SFC is the mass of fuel consumed per kilowatt-hour of electrical energy produced. Chiller plant efficiency (Figure 1.3b) is given in terms of the electric power required to produce 1 Ton of refrigeration (1 Ton = 12,000 Btu/hr). Marine chiller plant efficiency is highly dependent on the temperature of the entering condenser water temperature (ECWT). In Figure 1.3b, curves for ECWT of 70° F and 86° F are shown. It can be seen that the chilling plant is more effective when cooler water is available for the condenser. In marine applications, seawater is often used to cool the refrigerant in the condenser portion of the refrigeration loop.



(a) (b)

Figure 1.3: (a) Gas turbine generator specific fuel consumption versus percent of rated load [13], (b) chiller plant power demand versus percent of rated load [14]

In general, the behaviors of the generators and the chiller plants are the same; they are most efficient when operated at or near 100% of the rated load. Thus, proper selection and configuration of these components is critical to ensuring optimal operating efficiency.

Identifying shipboard energy system configurations that optimize the two performance objectives described above is a complex engineering challenge. There are a large number of design variables, and the range of possible solutions is large enough to preclude an exhaustive search of the solution space. Furthermore, a design change that improves one of the objectives may result in a decrease in performance for the other objectives. Therefore, a design framework is needed to help ship designers understand these trade-offs during the initial stages of ship design. This type of early-stage design tool would help reduce development costs and result in final ship designs with more efficient and survivable operating characteristics.

1.2 COMPLEX ENGINEERING SYSTEM DESIGN CHALLENGES

Design of shipboard distributed energy systems is a large, complex, multidisciplinary engineering problem. The AES energy system design problem embodies several characteristics that are typically encountered in complex engineering system design. These challenges include computational expense, compatibility of subsystem models, discrete variables / discontinuous objective functions, and complex multidisciplinary system interactions.

High-fidelity models of shipboard components and subsystems are typically too computationally expensive to enable rapid design or model-based reconfiguration. When performing design optimization or design space exploration, numerous evaluations of the objective function are often required to assess system performance as the design space is explored. This task can be made impossible if the computational expense of a single simulation is prohibitively long.

In addition to computational expense, a significant problem with multidisciplinary design problems is that subsystems and individual components may be modeled in disparate, non-compatible software programs [15,16]. For example, a commercial CFD package may be used to simulate hydrodynamic drag on a ship hull, while a different FEA package may be used to model stress and impact deformation. The shape of a ship hull has effects on both hydrodynamics and structural integrity, and a multidisciplinary analysis is thus required.

When analyzing system configurations or damage scenarios, many of the variables that define the system-level model are discrete and have a discontinuous effect on associated performance metrics. Examples include choices between different types of energy storage elements (batteries, flywheels, capacitors) and failure of individual components (*e.g.*, a generator or chilled water loop) in a damage scenario. This property

of a total-ship system model has significant implications on the types of algorithms that can be used for optimization and design space exploration.

Complex interactions exist between shipboard thermal and electrical systems that require a multidisciplinary, total-system model rather than stand-alone thermal or electrical models. Furthermore, a highly reconfigurable, searchable model with variables that can be easily manipulated is necessary to enable system-level design and configuration. Many commercially available modeling platforms do not share this property, and analysis of a new system architecture often requires an entirely new model to be constructed.

1.3 HYPOTHESIS AND RESEARCH OBJECTIVES

In this section, a hypothesis and supporting research objectives are introduced for addressing the complex engineering system design challenges described in the previous section. The primary contributions of this research can be captured by the following hypothesis:

> *Continuous variable metamodels and a discrete variable classifier-guided sampling method can be used as part of a system-level design framework to enable rapid design optimization of computationally expensive shipboard energy distribution system models.*

This hypothesis can be decomposed into four interrelated subhypothesis. The first subhypothesis is that metamodels can be used in place of computationally expensive models of shipboard energy distribution system components to facilitate rapid design and configuration studies. The second subhypothesis is that the classifier-guided sampling method can be used to supplement metamodels for efficient design exploration of computationally expensive, system-level, discrete variable/discontinuous response design problems. The third subhypothesis is that the highly reconfigurable representation of

shipboard energy systems that is presented earlier in this chapter will enable system-level design studies that yield useful early-stage design recommendations. Lastly, the fourth subhypothesis is that the achievements of the first three subhypotheses can be synthesized to create a system-level design framework for total-ship energy system design. In the following subsections, specific research tasks are described for addressing each of the four subhypotheses.

1.3.1 Task 1 – Scalability of Metamodels for Continuous Variables

Metamodels, also known as surrogate models, can be used in place of computationally expensive simulation models to increase computational efficiency for the purposes of design optimization or design space exploration. A metamodel is developed by fitting an approximation to a set of training points that are generated using an expensive base model. Metamodel-based design optimization is especially advantageous for ship design problems that require either computationally expensive simulations or costly physical experiments.

Many of the all-electric ship subsystem or component models may be computationally expensive. Furthermore, many of the simulation models required for even a simple system are constructed in different software programs, which makes it hard to integrate the models for the purpose of system-wide design exploration. Metamodels can be used as fast and accurate approximations of expensive computer models [17], and they can be built in a common programming environment [15,16]. Therefore, these two benefits of metamodels make them ideal for inclusion in the proposed design framework.

The performance of metamodeling techniques is well understood for functions in low to moderate dimensions (one to ten independent variables). However, very little has been published on the subject of metamodeling in high dimensions [18]. Furthermore,

many of the previously published comparison studies of metamodeling performance use specific test functions and models, and it is difficult to generalize the results of these studies to other functions. Therefore, the first task of this research is to perform a systematic study of the scalability of metamodeling techniques. The study uses a set of test functions that can be scaled to an arbitrary number of dimensions and exhibit varying types of behavior so that a better understanding of high-dimensional, generalized metamodel performance can be gained.

1.3.2 Task 2 – Classifier-Guided Sampling for Computationally Expensive, Discrete Variable / Discontinuous Response Functions

Metamodels are well-suited for approximating individual component models with continuous variables and responses, but they are not readily applicable to discrete or discontinuous systems [19,20]. For example, consider the problem of selecting an appropriate generator for an off-grid electrical system. The type of generator selected (diesel, gas turbine, *etc.*) is a discrete variable. The power rating of the generator is also a discrete variable, because generator capacities must be selected from commercially available standard sizes. Furthermore, the selection of the generator would have a discontinuous effect on overall system performance measures such as efficiency.

Metamodels are not suitable for these types of problems because they can only be used to approximate functions with continuous design variables and smooth, continuous responses. Accordingly, the second task of this research is to establish a novel sampling and search strategy to accompany the metamodels. The methodology will use a Bayesian classifier in place of a metamodel for evaluating system performance in the presence of discrete design variables and/or discontinuous responses. A classifier is similar to a metamodel in that it is used to predict the performance of new instances based on a set of known observations or training points. However, a classifier assigns an unevaluated test

point to a categorical class label rather than attempting to predict the resulting response in continuous space. The method to be developed exploits this property of classifiers by using it to guide a sampling process towards combinations of continuous and/or discrete design variable values with a high probability of yielding preferred performance.

1.3.3 Task 3 – Reconfigurable Total-Ship System Model

Significant effort has been dedicated to system-level shipboard energy system modeling. The Dynamic Thermal Modeling and Simulation Framework (DTMS) [21-23], developed at Applied Research Laboratories at UT Austin, is a powerful tool for modeling and simulating system-level dynamic thermal systems. The Virtual Test Bed (VTB), developed at the University of South Carolina, was initially capable of supporting "design, analysis, and virtual prototyping of electric power systems" [24]. VTB has since been updated to include thermal systems [25].

Both VTB and DTMS are useful tools for modeling and analyzing a specific thermal or electrical architecture, but they are not currently well suited for rapid design space exploration or real-time reconfiguration because they only allow "one at a time" model simulations. That is, a specific system architecture is "drawn" using the user interface, a simulation is run, and the result is presented to the user. The parameters of individual model components (*e.g.* generator sizes) can be easily manipulated. However, it may be desirable to evaluate completely different system architectures or configurations in which the types, locations, and connectedness of components are being investigated. In such cases, a completely new system model must be constructed to evaluate performance of the new candidate design. Therefore, the third task of this research is to use the multi-energy domain, zonal modeling approach that is described in Section 1.1 to develop a highly reconfigurable, parameterized, system-level model for

rapid design exploration and reconfiguration. In this research, MATLAB and Simulink are used for all component and system-level models, metamodels, and design tools.

1.3.4 Task 4 – System-Level Design Framework

The fourth task of the research presented in this dissertation is to develop a system-level design framework to perform design of total-ship energy distribution systems. The design framework synthesizes Tasks 1-3 described above to form an early-stage design tool for determining the optimal size, location, and distribution architecture of thermal and electrical system components. The highly reconfigurable total-ship energy system model serves as a platform for evaluating the performance of each candidate system design. The knowledge gained from the high-dimensional metamodeling scalability study provides insights as to how metamodels can be used to provide fast approximations of computationally expensive continuous variable / continuous response component and subsystem models. The classifier-guided sampling method, which is specifically developed for computationally expensive, discrete variable / discontinuous response problems, is used to identify promising system-level configurations while minimizing the number of required simulation runs.

1.4 ORGANIZATION

This chapter provides an overview of the motivation, hypothesis, and specific objectives of this research. In Chapter 2, a literature review of concepts related to complex engineering system design is provided. The concepts covered in the literature review include metamodeling techniques for continuous variables, metamodel based design for problems with discrete variables or continuous responses, and discrete variable optimization methods. In Chapter 3, a study is conducted to investigate the scalability of continuous variable metamodeling techniques in high dimension. The metamodeling

techniques are applied to a set of test functions with varying global behavior that can be scaled to an arbitrary number of dimensions. In Chapter 4, the classifier-guided sampling method is developed for solving discrete variable / discontinuous response design problems. The method is validated by comparing its performance to genetic algorithms on a set of discrete variable optimization problems. In Chapter 5, the computer implementation of the reconfigurable total-ship energy system model is described. In Chapter 6, metamodels and the CGS algorithm are applied to the reconfigurable ship model to identify ship configurations that consume less fuel and are more survivable. Lastly, conclusions and opportunities for future work are discussed in Chapter 7.

Chapter 2: Complex Engineering System Design Research

Three challenges that are frequently encountered in complex engineering system design are computational expense, subsystem model incompatibility, and the existence of discrete variables / discontinuous objective functions. In this chapter, existing methods that were developed to address these challenges are reviewed to provide the necessary context and motivation for the research tasks in this dissertation.

In Section 2.1, an overview is given of how metamodeling techniques are used in design optimization to alleviate computational expense. Furthermore, the field of multidisciplinary design optimization (MDO) is discussed, in which metamodels are often used to combine disparate subsystem models into an inexpensive, multidisciplinary, total-system representation.

In Section 2.2, published metamodeling comparison studies are reviewed that experimentally assess the performance of metamodeling techniques when applied to functions with a low to moderate number of independent variables.

A significant shortcoming of metamodels is that they are not applicable to discrete variable / discontinuous function design problems. In Section 2.3, two methods that have been developed in an attempt to address this issue are described.

Several optimization methods have been developed to handle discrete variable / discontinuous problems. However, these methods do not specifically address the issue of computational expense. These methods, which include simulated annealing, genetic algorithms, and tabu search, are reviewed in Section 2.4.

Lastly, the findings of this literature review, which include areas in which more research is warranted, are discussed in Section 2.5.

2.1 METAMODELING TECHNIQUES FOR ENGINEERING DESIGN

Computer simulations of physical systems are often complex and computationally expensive, requiring minutes or hours to complete a single simulation. While the accuracy and detail offered by a well-constructed simulation model are indispensable, the required execution time is unacceptable in certain situations. For example, the solution of an optimization problem requires numerous evaluations of an objective function as the design space is explored. If one or more lengthy simulations are required for each evaluation of the objective function, it may not be possible to identify a satisfactory design point in an acceptable amount of time.

Metamodels are used in place of the original simulation models to provide reasonable approximations in a fraction of the time. Metamodels are developed using a set of training points from the original simulation. Once built, the metamodel is used in place of the original simulation model to predict responses quickly and repeatedly. For example, the surface shown in Figure 2.1 is a metamodel built using the indicated training points.

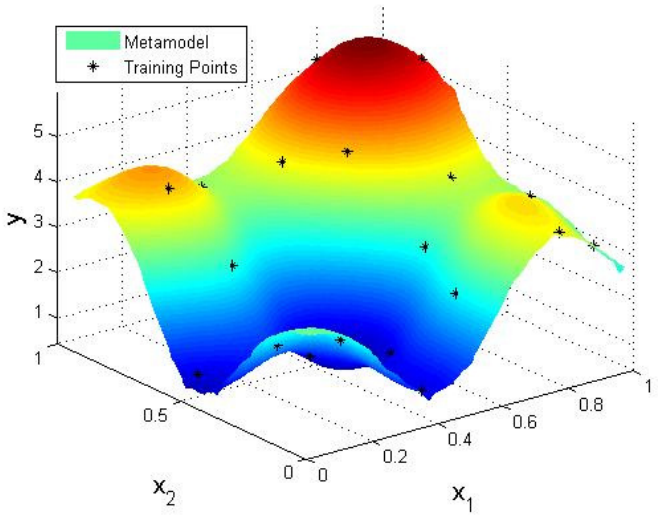


Figure 2.1: A two-dimensional metamodel of y as a function of x_1 and x_2.

The metamodeling methods that most frequently appear in the engineering design literature are polynomial regression (PR) [26], multivariate adaptive regression splines (MARS) [27], support vector regression (SVR) [28], kriging [29], radial basis functions (RBF) [30], and neural networks [31].

2.1.1 Metamodel-Based Design Optimization

Metamodeling techniques have numerous applications to engineering design. Most notably, metamodels are integrated with optimization algorithms to form what is known as metamodel-based design optimization (MBDO). In MBDO, metamodels are used to provide a fast approximation of a computationally expensive objective function. There are multiple ways in which metamodels are used in MBDO, most of which can be categorized as one of the three approaches described by Wang and Shan [19]: sequential, adaptive, and direct sampling (Figure 2.2).

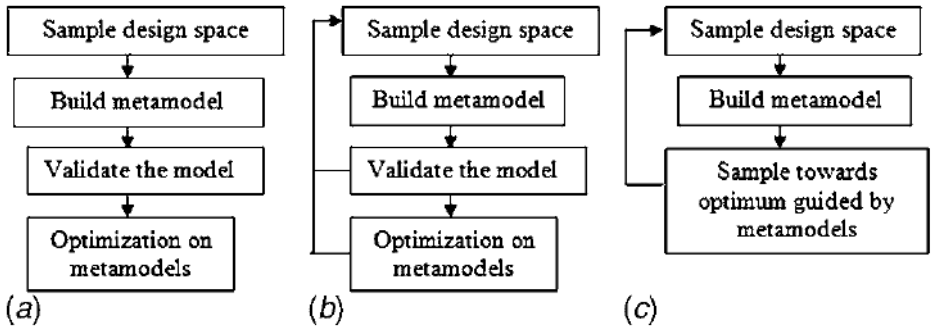


Figure 2.2: Metamodel based design optimization strategies [19]: (a) sequential approach, (b) adaptive approach, and (c) direct sampling approach.

The sequential approach (Figure 2.2a) uses a metamodel to perform optimization in four steps with no loops or iterations. First, the design space is sampled and the expensive objective function is evaluated. Second, the metamodel is constructed using the training points from the first step. Third, the metamodel is validated to ensure that the

metamodel is providing satisfactory accuracy for optimization purposes. An overview of metamodel assessment strategies, including metamodel validation, is provided by Meckesheimer *et al.* [32]. The final step in the sequential approach is to perform optimization on the inexpensive metamodel with the presumption that the optimal or near optimal solution obtained is similar to that which would be obtained if optimization were performed on the original, expensive objective function model.

The adaptive approach (Figure 2.2b) is similar to the sequential approach described above; the main difference is that the process is performed in a loop such that further expensive samples can be taken if the metamodel fails some user-defined validation or optimization criteria. Examples of methods that use this approach are the Surrogate Management Framework [33] and the total management framework [34].

The direct sampling approach (Figure 2.2c) is significantly different from the sequential and adaptive approaches. The direct sampling approach uses the metamodel to guide the sampling process towards optimal solutions as opposed to performing optimization on the metamodel. The Mode-Pursuing Sampling (MPS) method [35] is an example of a direct sampling metamodeling approach. The MPS method uses a metamodel to rank a large set of candidate solutions based on their supposed objective function values, as indicated by the metamodel. A set of points are selected from this large set of so-called "cheap points", and selection is performed such that solutions with preferred metamodel objective function values have a higher likelihood of being selected.

The key difference between the first two approaches (sequential and adaptive) and the third approach (direct sampling) is that in the first two approaches, optimization is performed using the metamodel as a substitute for the objective function. With the direct sampling approach, the metamodel is only used to guide the sampling process towards optimal solutions.

MBDO has been applied to a large variety of problem types, including vehicle impact (crashworthiness) optimization [36], piezoelectric device design [37], structural design optimization [38], and aerodynamic shape design [39].

2.1.2 Multidisciplinary Design Optimization

Another area in which metamodels are commonly applied is multidisciplinary design optimization (MDO). Broadly, MDO is a "methodology for the design of systems in which strong interaction between disciplines motivates designers to simultaneously manipulate variables in several disciplines" [15]. By this definition, MDO is the process of optimizing complex engineering systems in which multiple, interacting, multidisciplinary subsystems exist.

Metamodels can play a critical role in MDO in several ways. Computational expense is one of the primary challenges in MDO [15], and metamodels can be used to mitigate this challenge when subsystem models are expensive. For example, Target Cascading is an MDO approach in which system-level performance targets are propagated to subsystem-level design targets. In this process, inexpensive subsystem models are essential, and expensive subsystem models should be replaced by inexpensive surrogates [40].

Multidisciplinary subsystem models are often built in different modeling environments and/or by different groups of designers [16]. In such cases, metamodels can be built in a common programming environment such as MATLAB or C++ and used to combine models from disparate software packages and modeling groups into a single total-system representation.

Lastly, when collaborative design is being performed in parallel by multiple groups of designers, design teams can access continuously updated system and subsystem

metamodel approximations to evaluate the effects of their individual design changes on overall system performance (*cf.* [41]).

2.2 METAMODELING COMPARISON STUDIES

The best metamodeling method for a particular application depends on the needs of the project and the nature of the expensive model that is to be approximated. The three most common criteria for evaluating metamodels include:

- *Accuracy:* Capability of predicting new points that closely match those generated by the base model.
- *Training Speed*: Time to build the metamodel with training data from the base model.
- *Prediction Speed*: Time to predict new points using the constructed metamodel.

Although several studies have compared the performance of alternative metamodeling methods, no single method has emerged as universally dominant. Rather, individual techniques have strengths and weaknesses and selection of the most appropriate method is dependent on several factors such as the complexity of the response function, the availability of training data, and the number of input variables. Most of the studies that have been conducted investigate the performance of metamodeling techniques when they are applied to functions with a low (1D - 5D) and moderate (6D - 10D) number of independent variables.

For example, Giunta and Watson [42] apply kriging and polynomial regression to quasi-sinusoidal and quasi-quadratic test problems with one, five, and ten design variables and evaluate the methods based on prediction accuracy. The five-dimensional and ten-dimensional functions are created by performing linear combinations of the one-dimensional function. In the one-variable test problem, the results of this study show that kriging outperforms polynomial regression for a sinusoidal test function. However, the

polynomial response surface predicts new points more accurately for a quadratic function in the one dimensional problem. This result is not surprising, because a quadratic polynomial would be expected to predict a quadratic unimodal function reasonably well. In higher dimensions, polynomial regression is found to be superior to kriging in both the sinusoidal and quadratic test functions, particularly in the quadratic case. This result implies that kriging may not be ideal for metamodeling high-dimensional multimodal problems. It should be noted that only two very specific functions were studied in this paper, and the conclusions should be extended to other types of test functions cautiously.

Wang *et al.* [43] evaluate the accuracy of MARS and four different polynomial regression methods (linear, piecewise linear, quadratic, and higher order) with varying quantities of training points. They apply each method to a five variable car frame stiffness problem and evaluate accuracy. They find that all methods perform similarly when few training points are available, but MARS and piecewise linear regression methods emerge as superior only when a large set of training data is available.

Jin *et al.* [44] apply polynomial regression, MARS, radial basis functions (RBF) and kriging to fourteen test problems with varying levels of nonlinearity and scales ranging from two to sixteen input variables. Overall, RBF is shown to be superior in terms of accuracy, robustness to various sample sizes, and ability to accommodate a large variety of problem types. Kriging is shown to be very competitive with RBF with respect to average and maximum error metrics compiled over the entire set of test problems, but it is found to be very slow compared to the other methods in terms of both time to build a new model and time to predict new points.

Clarke *et al.* [45] apply support vector regression (SVR), polynomial regression, kriging, RBF, and MARS to a set of 26 test functions with dimensionality ranging from two to four variables. The performance of each method is evaluated on the criteria of

accuracy, speed and robustness. The results indicate that SVR has the lowest level of average error when applied to linear and non-linear problems. However, kriging has slightly less maximum error than SVR. This study indicates that SVR is an excellent method for developing metamodels for either linear or non-linear, low-dimensional (two to four dimensional) problems.

Fang *et al.* [36] apply polynomial regression and RBF to a multi-objective crashworthiness problem. The study includes ten design variables and three different objective functions. The study indicates that RBF is more computationally expensive than polynomial regression but provides more accurate results for more nonlinear objective functions.

Lee *et al.* [46] propose a method for using SVR to solve optimization problems. Their method involves repeatedly reforming the metamodel for increasingly focused regions of the design space as the optimal solution is approached. In all five of the test problems, SVR finds the known optimal solution with the least number of iterations when compared to kriging, RBF, and polynomial regression. These results indicate the superior accuracy and efficiency of SVR. The problems in this study are highly nonlinear, and therefore it can be observed that SVR handles nonlinear problems well as it dips and rises into and out of local minima and maxima without under-fitting. This study shows that SVR is an effective method for approximating and solving nonlinear, multimodal optimization problems.

Kim *et al.* [47] use moving least squares (MLS) regression, kriging, RBF, and SVR to approximate a set of mathematical functions with dimensions of two, four, six, and eight variables. All of the functions are highly nonlinear, and some of them are multimodal. The methods are evaluated based on accuracy. Results show that the kriging and MLS models perform the best, particularly in the moderate dimensional (6 or

8 variable) test problems. The results also show that prediction error increases significantly for SVR and RBF as the number of dimensions increases. This pattern suggests caution must be used when applying SVR and RBF to moderate-dimensional problems to ensure that proper tuning parameters are used and that an adequately sized set of training data is available.

Ely and Seepersad [48] apply polynomial regression, MARS, kriging, and SVR to a four-dimensional welding problem. The four methods are compared and analyzed with respect to efficiency and accuracy metrics. The modeled response, which is temperature on a flat welded plate, resembles a highly non-linear, unimodal function. By looking at cross sections at varying distances from the weld zone, temperature gradients of decreasing steepness are observed with the steepest temperature gradients nearest the weld zone. Careful analysis reveals that kriging is most accurate near the weld zone, while SVR is most accurate farther away from the weld zone. Due to the sharp nature of the response near the weld zone, the method that passes through each training point provides the most accurate fit in that region. Time metrics indicate that SVR is fastest in both build time and prediction time.

Based on the literature reviewed in this section, several conclusions can be made with regard to scalability and multimodality of the reviewed metamodeling methods. In general, polynomial regression models can be built and run very quickly. However, polynomial response surfaces are unable to predict highly non-linear and multimodal functions in multiple dimensions. On the other hand, kriging and radial basis functions are both capable of modeling nonlinear and multimodal functions with higher computation time than polynomial regression. Multivariate adaptive regression splines are capable of modeling multimodal functions in moderate-dimensional space, but often require large training data sets and are computationally expensive to build. Support

vector regression, which appears to be the most promising method reviewed here, is shown to be capable of modeling low and moderate-dimensional multimodal functions accurately with minimal computational expense.

While these trade-offs are relatively well understood for low-dimensional problems, most comparative studies have been limited to ten or fewer input variables, and it is unclear how the behavior of different metamodels scales to higher dimensions.

2.3 METAMODELS FOR DISCRETE VARIABLES AND DISCONTINUOUS FUNCTIONS

Metamodeling has been studied extensively for problems with continuous variables. However, experts in the field recognize that applying metamodeling techniques to problems with discrete or mixed discrete/continuous variables is an ongoing challenge that is yet to be addressed. For example, in a 2007 review of metamodel-based design optimization methods, Wang and Shan [19] assert that "…studies on metamodels and metamodeling techniques for problems with mixed discrete and continuous variables are lacking." Furthermore, in a 2008 review of metamodeling methods in multidisciplinary design optimization, Simpson *et al.* [20] note that "Challenges with handling mixed discrete/continuous variables still exist, and also may have gotten worse due to the nature of the problems now being investigated." Lastly, in a 2009 review of recent advances in metamodel-based optimization, Forrester and Keane [49] explicitly state that "The first assumption we make with all the surrogate modelling techniques discussed here is that the engineering function is continuous".

Huang and Arora [50] identify five types of discrete variable optimization problems. In the least severe problem type, variables are restricted to discrete values but the objective function is continuous and differentiable. For example, the bending stress in a rectangular beam is a continuous function of the cross sectional dimensions, but the

designer may be forced to choose from a finite set of commercially available sizes. In more severe discrete variable problem types, the variables are categorical and can only take discrete values because the objective function is otherwise undefined. This situation arises when the variables are categorical such as material type or cross section style (I-beam, C-channel, angle, *etc.*).

Metamodels can be used to solve design problems with continuous or discrete variables, as long as the assumption of objective function continuity holds true. However, computationally expensive engineering design problems that feature discrete variables and discontinuous responses are common, and a traditional metamodel cannot be used as a global approximation of total system behavior in such cases. For example, in marine engineering, ship hull designs can take many shapes and may or may not have additional features such as a front bulb, stabilizers, or bilge keels. In many cases, computational fluid dynamics (CFD) models are used as tools to analyze performance characteristics of candidate designs such as ship speed, stability, and hydrodynamic drag. In structural optimization, member shapes, materials, and arrangements are all described by discrete variables, and performance criteria such as stress or stiffness are evaluated using computationally expensive finite element models.

Two notable methods have been developed for discrete variable / discontinuous function metamodel-based design. The first method, developed by Meckesheimer *et al.* [51], is a metamodeling approach for approximating problems with continuous variables and a response that has combined discontinuous/continuous behavior. It is assumed that the user has advance knowledge that the function to be approximated has two or more contiguous regions of continuity that are discontinuous at their boundaries. A single metamodel cannot be used to provide a global approximation of this type of function because training points that lie in different regions of continuity cannot be interpolated as

if part of a smooth function. This single continuous metamodel approach is shown in Figure 2.3, where $f(x)$ is the actual function, and $\phi(x)$ is the metamodel.

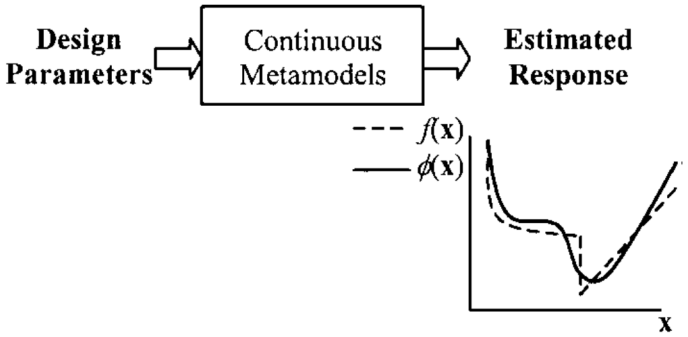


Figure 2.3: Single global metamodel applied to a discontinuous function [51]

From the figure, it can be seen that the metamodel approximation is very poor in the region of the discontinuity. To address the shortcoming, a method is developed that uses multiple metamodels to approximate the function, where each continuous region has its own metamodel approximation. A "logic function" is used to determined which metamodel to use based on the value of the input variable. The method is illustrated in Figure 2.4, where $s(x)$ is a logic function that selects the appropriate region of the design space and therefore the correct metamodel to be used based on the value of x.

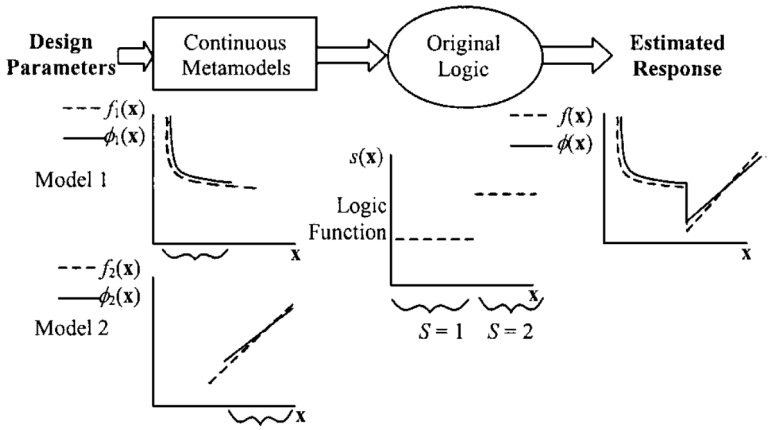


Figure 2.4: Continuous variable function with mixed discontinuous/continuous response [51]

This method is a function approximation approach and does not include a mechanism for exploring the design space in search of improved solutions. Furthermore, it is only applicable to a specific type of function in which there are multiple contiguous regions of continuity separated by discontinuities.

The second method reviewed here is the discrete-variable mode pursuing sampling method (D-MPS), developed by Sharif *et al.* [52]. Rather than attempting to build a metamodel that accurately represents the entire design space, the D-MPS method reduces expensive base function evaluations by using a metamodel to guide the search process towards regions of the design space where good solutions are believed to exist. This method can be categorized as a "Direct Sampling" MBDO approach, as described in Section 2.1. The method is performed in three main steps:

Step 1: "Generating Cheap Points" – A large set of candidate solutions is generated by uniformly sampling the design space. From that set, a smaller set is selected for expensive base function evaluations. A linear spline metamodel is constructed using these points and their corresponding outputs from the expensive base model.

Step 2: "Approximation" – All of the candidate solutions are evaluated using the current metamodel approximation. *The appropriateness of using a linear spline or any traditional metamodel relies on the assumption of a continuous, twice-differentiable base function.*

Step 3: "Discriminative Sampling" – The cheap metamodel evaluations are ordered based on their fitness as estimated by the metamodel. The points are then normalized thus creating an analogous probability density (Figure 2.5a). A cumulative distribution function (CDF) is generated from this probability density

(Figure 2.5b). A set of random numbers, generated on the interval [0, 1], is used to select points from the CDF using inverse transform sampling. Given the shape of the CDF, better solutions have a higher likelihood of being sampled.

This process is repeated until a predetermined termination criterion is met. This approach ensures that points that are believed by the metamodel to result in good objective function values are more likely to be sampled for subsequent expensive base model evaluation, thus reducing the total computational expense of the search process.

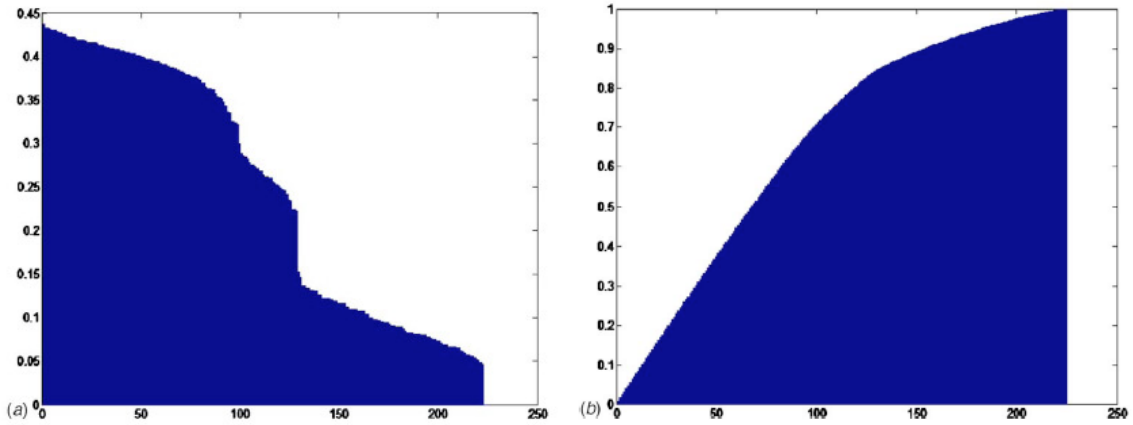


Figure 2.5: (a) Plot of the ordered and normalized "cheap" points, (b) cumulative distribution function used for further sampling of points for expensive base function evaluation [52]

The authors [52] use the D-MPS method to successfully solve three discrete variable test problems. However, all of them are functions that are twice continuously differentiable. That is, even though the variables are restricted to discrete values, the underlying functions are smooth and continuous. Although the D-MPS does not rely on function differentiation in the search process, a continuous differentiable base function is an essential requirement for the application of this method. A traditional metamodel is used to provide a global approximation of the base function in Step 2 listed above.

Therefore, this approach is not appropriate for problems in which the response is discontinuous and cannot be globally approximated with a traditional metamodel.

2.4 DISCRETE VARIABLE OPTIMIZATION

In this section, three well established optimization methods that are capable of solving discrete variable, discontinuous optimization problems are reviewed. Methods reviewed include simulated annealing, genetic algorithms, and tabu search.

2.4.1 Simulated Annealing

Simulated annealing (SA) [53] is a randomized search approach that is based on the mathematical representation of metallurgical annealing. During the search process, SA always accepts improving moves but only sometimes rejects moves that yield a less preferred solution, thus avoiding fixation on local minima. The probability of a non-improving, or "uphill" move being accepted is given by:

$$P = \exp\left(-\delta / T\right) \tag{2.1}$$

where δ is the difference between the current best and new solution (unfavorable change amount), and T is a user-defined control parameter that is analogous to temperature. In Figure 2.6, the relationship between P, T, and δ is illustrated:

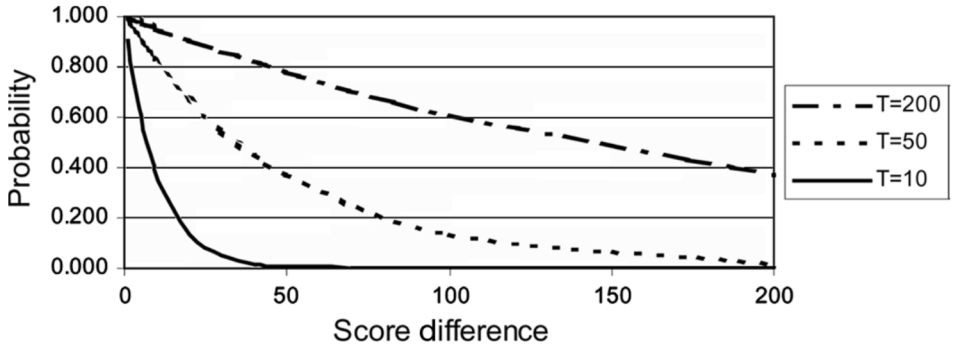


Figure 2.6: Simulated annealing acceptance probability function [54]

From Figure 2.6, it can be seen that the probability of an uphill move being accepted decreases with an increasing value of δ. The temperature T has an overall effect on the shape of the probability curve, and decreasing values of T decreases the average probability that an uphill move will be accepted. Furthermore, the value of T is decreased systematically throughout the solutions process, so that as "time" passes the probability of accepting an uphill move decreases. The process terminates when the value of T reaches zero or when the solution obtained at each T remains unchanged for a specified number of consecutive temperature changes [55]. If the temperature were to become zero, simulated annealing would always move to a neighboring solution that is better than the current best solution and could therefore terminate in a local minimum. It is the cooling schedule based probability of acceptance that gives simulated annealing its ability to move uphill and explore other regions of the design space. It has been shown that as the cooling schedule is extended to infinity, the probability of converging on the global optimum approaches 100% [56].

Simulated annealing has been successfully applied to many discrete variable optimization problems. For example, van Laarhoven *et al.* [57] successfully apply simulated annealing to job shop scheduling problems. Simulated annealing has also be applied classic operations research problems such as the traveling salesperson problem [58], the vehicle routing problem [59], and the bin packing problem [60]. The method has also been applied to the optimization of heat exchanger networks [61]. Simulated annealing has also been successfully applied to multimodal problems with continuous variables [62].

2.4.2 Genetic Algorithms

Genetic algorithms (GA) [63] are a class of solution methods that are designed to be analogous to natural evolution. There are four basic steps in the GA solution process: initialization, selection, reproduction, and termination [64]. In the initialization phase a population of candidate solutions is generated. Initialization concludes with each candidate solution being evaluated for fitness. During the selection phase, candidates from the initial population are selected to go into a "mating" pool. This is usually done probabilistically so that the fittest candidates are most likely to be selected. During reproduction, selected candidates are paired for mating. Mating may be done in a variety of ways, but a common mating maneuver is called crossover, in which two candidate solutions will split and trade a specified portion of their solution string. During this phase, a mutation operator is often implemented that forces a small change to one or more offspring that result from crossover. The process of selection and mating is repeated, thus creating a new "generation" of candidate solutions with each new iteration. Lastly, termination occurs when an acceptable solution is obtained, a fixed number of generations have been created, or several successive iterations have occurred negligible improvements in solution fitness.

Genetic algorithms represent a method that includes randomization in its procedure, but is not a purely random process. Certain steps of the process are randomized, such as the generation of the initial population. However, GAs do not proceed to a solution without guidance and direction. For example, during the selection phase, candidates are chosen at random, but the fittest solutions have a much high probability of being selected than the less fit ones. It is the use of a population of solutions, rather than one solution at a time, which gives GAs the ability to search areas of the design space that are outside of the current local optimum.

Genetic algorithms have been successfully applied to numerous types of problems in and outside of engineering. In the operations research domain, Chen *et al.* [65] apply GAs to discrete variable flow shop problems and found that they converge on the global optimum in a reasonable amount of time. In a mechanical engineering design application, Kumar *et al.* [66] apply GAs to the optimal design of crowned cylindrical roller bearings. In this case, the objective is to increase the life of the bearing by reducing fatigue stress in the rollers. The variables and constraints are all continuous variable geometric parameters.

Genetic algorithms have also found applications in automotive engineering. A common problem in automobile design is the design of the suspension system. The objective is to choose the spring stiffness and damping coefficients of the suspensions so that the maximum absolute acceleration that is experienced by the passengers is minimized. This acceleration is highly nonlinear and is dictated by the solution of a second order differential equation. Alkhatib *et al.* [67] successfully apply GAs to this type of problem, while Baumal *et al.* [68] also solve this type of problem with the added control variable of an active suspension system.

2.4.3 Tabu Search

Tabu search (TS) [69] is a solution approach that exploits the use of efficient computer data structures to search the design space intelligently. There is typically no randomization in the search procedure. Rather, TS makes moves according to user-defined rules about how to proceed to neighboring solutions. For example, in a discrete variable combinatorial optimization problem, the neighborhood may be defined as sites that can be reached by swapping two adjacent elements of a solutions string. Moves that have been made recently are designated and stored in a data structure as "tabu"

(forbidden) so that those moves will not be made again for some number of iterations. This concept is called recency based memory. A move that is classified as tabu is called "tabu active". This approach forces the solution procedure to try different kinds of moves rather than continuously repeating those that yield the most immediate gain [70].

The concept of recency based memory is only one way that TS methods use memory to search the solution space. Others include frequency, quality, and influence. Furthermore, mechanisms can be implemented that help avoid getting stuck in a local minimum (looping), and that can override tabu status if the tabu move yields a large improvement in solution quality (aspiration criteria) [71].

Tabu search is primarily used for solving combinatorial optimization problems but can also be used to solve problems with continuous variables [72]. Application of tabu search is difficult because the memory structures and neighborhood search strategies are highly problem dependent. A thorough understanding of the problem to be solved is essential to the successful implementation of tabu search.

Tabu search has been successfully applied to many classic discrete variable operations research problems. For example, the vehicle routing problem with time windows has the constraint that the fleet vehicles must arrive at and depart from a location in a certain time window. Tan *et al.* [73] successfully solve this type of problem with tabu search. In a similar application, Korsvik and Fagerholt [74] solve a ship routing problem where the vessels have flexible cargo capacities. Although the previous two examples are for combinatorial problems with integer variables, tabu search is not strictly limited to problems of this type. Siarry and Berthiau [72] develop a modification of tabu search, in which the neighborhood of a current solution and the tabu list structure is defined for continuous variable problems.

2.5 LITERATURE REVIEW FINDINGS

The metamodeling comparison studies reviewed in this chapter indicate that the performance of traditional continuous variable metamodeling techniques in low to moderate dimensions is well understood. However, optimization problems often have large numbers of input variables. For example, a model of a marine vehicle hull form has up to 150 geometric parameters that describe its shape [75]. Performance metrics such as hydrodynamic drag are evaluated using computationally expensive CFD models and the use of high-dimensional metamodels would enable rapid optimization and design space exploration. Sometimes, it is possible to use screening methods to reduce the number of input variables [76], but the methods must be used with caution. For some optimization problems, unimportant input variables in one region of the design space may be highly influential in other nearby regions. Accordingly, it is not always possible to reduce the number of variables using screening methods and it is important to understand the behavior of metamodels as they are scaled to higher dimensions. Very little has been published on the subject of metamodeling in high dimensions [18]. Therefore, a systematic study of the scalability of metamodeling techniques is needed to gain an understanding of their performance in higher dimensions. This task is performed in Chapter 3 of this dissertation.

A significant shortcoming of traditional metamodeling techniques is that the function being approximated must be smooth and continuous. In many cases, systems design problems resemble combinatorial optimization problems that have discrete variables and a discontinuous objective function. Two metamodel based approaches that partially address these issues are reviewed in this chapter. The first method [51] is only appropriate for combined continuous / discontinuous objective functions, while the

second method [52] is only appropriate for discrete variable optimization problems with continuous objective functions.

The discrete variable optimization techniques reviewed (simulated annealing, genetic algorithms, and tabu search) are all "direct search" optimization techniques. Contrary to traditional optimization techniques that rely on an explicit mathematical representation of the objective function, direct search optimization techniques are guided only by objective function values at previously evaluated design points. Therefore, they are capable of solving "black box" problems and require no explicit knowledge of the objective function. The downside of direct search methods is that the objective function must be evaluated for each new candidate solution throughout the search. The total number of function evaluations can be quite large, and solution time may be prohibitively long if the objective function must be evaluated using a computationally expensive model or simulation.

For the reasons described above, a method for solving computationally expensive, discrete variable, discontinuous objective function design problems is needed. In Chapter 4 of this dissertation, a classifier-guided sampling method that is appropriate for these types of problems is developed and validated.

Chapter 3: Metamodel Comparison for Continuous Problems

The published comparison studies reviewed in the previous chapter show that the performance of alternative metamodeling techniques in low and moderate dimensions is well understood. However, little has been done to investigate the performance of these methods when applied to problems with more than ten variables. Several studies [42,44,47] show that performance of different metamodeling techniques changes with the scale of the problem, suggesting that it is important to investigate these effects further. The test functions used to evaluate metamodeling performance in the comparison studies are very specific, and the results of these studies cannot necessarily be used to make general conclusions about the performance of metamodeling techniques when applied to broad classes of problems.

In this chapter, a selection of metamodeling techniques are applied to three distinct test functions: a kernel density function, the Rosenbrock function, and a modified version of the Rosenbrock function. The test functions are scaled incrementally from 15 to 50 independent variables to gain an understanding of how well metamodeling techniques perform when applied to functions with a large number of independent variables. The test functions exhibit two characteristics that are important for this study. First, each of the test functions can be scaled with respect to the number of input variables, making it possible to observe trends in metamodeling performance on common underlying functions with increasing numbers of input variables. Second, the three test functions are designed to represent broad classes of engineering application problems, ranging from problems with various levels of modality (kernel density functions), to problems with sharp gradients (Rosenbrock function), to problems with relatively smooth gradients throughout (modified Rosenbrock function).

3.1 METAMODELING METHODS SELECTED FOR STUDY

Three of the most popular metamodeling techniques for engineering design applications are compared in this study: kriging, radial basis functions, and SVR. Polynomial response surfaces are also popular, but they are not considered because they are difficult to construct for high-dimensional, multimodal functions, with the exploding number of possible interaction terms available for inclusion in the final model. Although MARS has been included in several comparison studies, it is not included in this study because it has been shown to require a relatively large set of training data to model multimodal functions [43,44,77]. The remainder of this section provides the mathematical detail of the three metamodeling techniques compared in this study.

3.1.1 Kriging

Kriging [29] consists of a combination of a known global function plus departures from the base model:

$$\hat{y}(\mathbf{x}) = \sum_{i=1}^{k} \beta_i f_i(\mathbf{x}) + Z(\mathbf{x}) \tag{3.1}$$

where β_i are unknown coefficients and the $f_i(\mathbf{x})$*'s* are pre-specified functions (usually polynomials). $Z(\mathbf{x})$ provides departures from the underlying function so as to interpolate the training points and is the realization of a stochastic process with a mean of zero, variance of σ^2, and nonzero covariance of the form

$$\operatorname{cov}\left[Z(x_i), Z(x_j) \right] = \sigma^2 R(x_i, x_j) \tag{3.2}$$

where R is the correlation function specified by the user. In this study, a constant term is used for $f(\mathbf{x})$ and a Gaussian curve of the following form is used for the correlation function:

$$R\left(x_i, x_j\right) = \exp\left[-\theta_i \left(x_i - x_j\right)^2\right] \tag{3.3}$$

where the θ_i terms are unknown correlation parameters that are determined as part of the model fitting process.

3.1.2 Radial Basis Functions

Radial basis functions [78] are an interpolation method that uses a linear combination of weights and basis functions whose values depend only on their distance from a given center, $\mathbf{x}_i$. Typically, a radial basis function metamodel takes the following form:

$$\hat{y}(\mathbf{x}) = \sum_{i=1}^{k} w_i \phi_i\left(\mathbf{x}, \mathbf{x}_i\right) \tag{3.4}$$

where the w_i is the weight of the i_{th} basis function, $\phi_i(\mathbf{x},\mathbf{x}_i)$. In this study, a Gaussian basis function of the following form is used to develop the metamodels for testing:

$$\phi\left(\mathbf{x}, \mathbf{x}_i\right) = \exp\left[-k\left\|\mathbf{x} - \mathbf{x}_i\right\|^2\right] \tag{3.5}$$

where k is a user specified tuning parameter.

3.1.3 Support Vector Regression

In support vector regression [28], the metamodel takes the form:

$$\hat{y}(\mathbf{x}) = \sum_{i=1}^{l} \left(\alpha_i - \alpha_i^*\right) k\left(\mathbf{x}, \mathbf{x}_i\right) + b \tag{3.6}$$

where the α terms are Lagrange multipliers, $k(\mathbf{x},\mathbf{x}_i)$ is a user specified kernel function, and b is the intercept. The α terms are obtained by solving the following dual form optimization problem:

$$\text{Maximize} \begin{cases} -\frac{1}{2}\sum_{i,j=1}^{l}\left(\alpha_i - \alpha_i^*\right)\left(\alpha_j - \alpha_j^*\right)k\left(x_i, x_j\right) \\ -\varepsilon\sum_{i=1}^{l}\left(\alpha_i + \alpha_i^*\right) + \sum_{i=1}^{l} y_i\left(\alpha_i - \alpha_i^*\right) \end{cases} \tag{3.7}$$

$$\text{Subject to} \begin{cases} \sum_{i=1}^{l}\left(\alpha_i - \alpha_i^*\right) = 0 \\ \alpha_i, \alpha_i^* \in [0, C] \end{cases} \tag{3.8}$$

In Equations (3.7) and (3.8), l is the number of training points, ε is a user-defined error tolerance, and C is a cost parameter that determines the trade-off between the flatness of the $\hat{y}$ and the tolerance to deviations larger than ε. In this study, a Gaussian kernel function of the following form is used to construct the metamodels for testing:

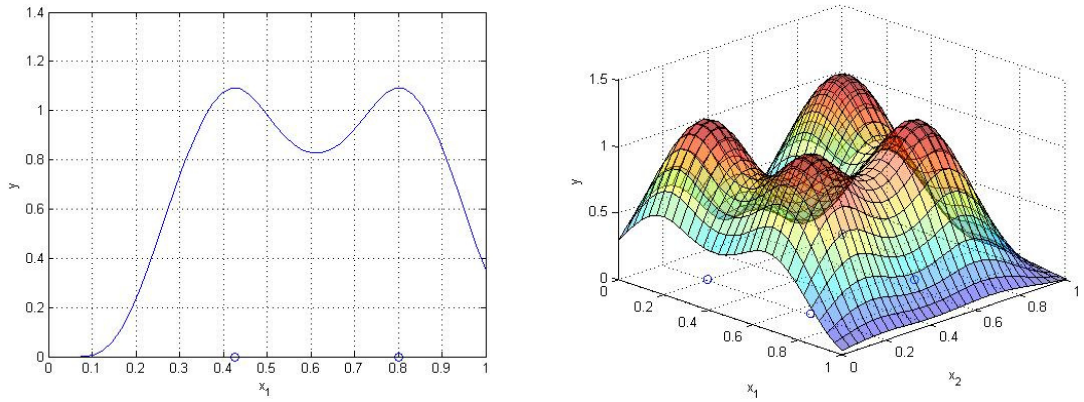
$$k\left(\mathbf{x}, \mathbf{x}_i\right) = \exp\left[-g\left(\mathbf{x} - \mathbf{x}_i\right)^2\right] \tag{3.9}$$

where g is a user specified tuning parameter.

3.2 TEST FUNCTIONS

3.2.1 Kernel Density Estimation

The kernel density estimation (KDE) method offers a way to generate test functions in any number of dimensions (D) that contain any number of kernels (N), where the number of kernels can be used to control the modality (number of local maxima or minima) of the resulting hyper-surface. For example, the functions in Figure 3.1 were generated using the KDE method:



(a) (b)

Figure 3.1: KDE functions with (a) $D = 1$ and $N = 2$ and (b) $D = 2$ and $N = 4$.

The KDE, also known as a Parzen window [79], is formulated as an average of N kernel functions, K, in product form [80]:

$$f(\mathbf{x}) = \frac{1}{N\prod_{i=1}^{D} h_i} \sum_{j=1}^{N} \left[\prod_{i=1}^{D} K\left(\frac{x_i - x_i^j}{h_i} \right) \right] \tag{3.10}$$

where D is the dimensionality of the problem. The shape of the KDE is controlled by the kernel function, the kernel centers, x_i^j, and the smoothing parameters, h_i. For this research, the tri-weight kernel function [80], shown in Equation (3.11), was selected for its smoothness and for the fact that it is not a Gaussian function, which was used as a basis function for some of the metamodeling techniques studied.

$$K\left(\frac{x_i - x_i^j}{h_i} \right) = \frac{35}{32} \left[1 - \left(\frac{x_i - x_i^j}{h_i} \right)^2 \right]^3 \tag{3.11}$$

for $x_i - x_i^j \le h_i$ and $K = 0$ otherwise

It is a fairly simple task to create a kernel function of arbitrary dimensionality. However, controlling the modality is a bigger challenge because careful consideration of the kernel centers and smoothing parameters is necessary. For the choice of kernel

centers, a certain amount of randomness is desired when the resulting KDE function is to be used as a test function for metamodeling comparison. The test function should be unique and its global behavior should be unknown when generating metamodel training points. However, a certain amount of control over placement of the kernel centers is needed for creating the required number of distinct local maxima and distributing them throughout the design space. This challenge was met by choosing the first kernel randomly from a uniform distribution over the input space and then choosing the remaining kernel centers sequentially such that the next center, $\mathbf{x}^{N+1}$, is the minimum of the current KDE which only includes the previous N center points:

$$\mathbf{x}^{N+1} = \arg\min \left\{ \frac{1}{N\prod_{i=1}^{D} h_i} \sum_{j=1}^{N} \prod_{i=1}^{D} \frac{35}{32} \left[1 - \left(\frac{x_i - x_i^j}{h_i} \right)^2 \right]^3 \right\} \qquad (3.13)$$

$$\text{Subject to: } \mathbf{x}_{\min} \le \mathbf{x} \le \mathbf{x}_{\max}$$

where $\mathbf{x}_{\min}$ and $\mathbf{x}_{\max}$ are upper and lower boundaries of all kernel locations. Using this approach, the next kernel center is placed at the location of minimum density, and the resulting sequence of kernel centers fills the space approximately uniformly. Equation (3.13) is solved using multi-start sequential quadratic programming (fmincon in MATLAB®), stopping if more than one second elapses or less than 1% improvement occurs for 5 consecutive sequential quadratic programming iterations. This procedure is not guaranteed to find the global minimum. However, finding the global minimum is not imperative; it is only necessary to find a point in a low density region.

As for the choice of smoothing parameters, h_i, setting them too small will result in a surface with steep gradients at each kernel center, while setting them too large results in overly smooth gradients and poor definition of the local maxima. However, for the case of the tri-weight kernel function, one can guarantee an N-modal function by setting the

smoothing parameter to 95% of the minimum Euclidean distance between any two kernel centers.

The KDE method is used to create four multimodal functions that are scaled from 15 to 50 dimensions (i.e., 15 to 50 independent variables). In the first scenario, there is $N = 1$ kernel (mode) regardless of the dimensionality of the problem. In the second scenario, there are $N = 2$ kernels, *i.e.* two local maxima regardless of the dimensionality of the problem. In the third and fourth scenario, there are $N = 5$ and $N = 10$ kernels, respectively. Five different KDEs are created for each combination of dimensionality and modality because of the random placement of the initial kernel in the KDE build process. The use of multiple KDEs indicates whether the performance of each metamodeling method is sensitive to the locations of the kernels.

The kernel density test functions create a unique challenge for the metamodeling methods. Specifically, the effect of scaling the number of input variables can be investigated independently for different levels of modality. Also, the kernel density functions represent a variety of different engineering problems. For example, it is common to encounter similarly shaped functions when modeling forced convection in heat exchangers with multiple passageways, such as prismatic cellular materials (*cf.* [81,82]). Exiting velocity and temperature profiles of cooling fluids tend to assume the multimodal form represented by the kernel density functions, where the number of modes corresponds to the number of passageways. As another example, bistable structures are characterized as structures that "snap" back and forth between multiple stable positions (*cf.* [83]). Those stable positions are characterized as energy wells in a plot of potential energy versus displacement, resulting in energy-displacement plots that closely resemble the multi-modal kernel functions used in this study. In both of these examples, the multimodal functions can be dependent on a large number of independent variables,

including a large number of dimensions, material properties, fluid properties, and other factors. Similar multimodal problems exist in structural dynamics, crash-worthiness, and other applications.

3.2.2 Rosenbrock Function

The Rosenbrock function is commonly used to test the performance of optimization algorithms [84-86]. The general form of the Rosenbrock function is given in Equation (3.14):

$$f(\mathbf{x}) = \sum_{n=1}^{D-1}\left[100\left(x_{n+1} - x_n^2\right)^2 + \left(x_n - 1\right)^2\right], \qquad -2 \le x_n \le 2 \tag{3.14}$$

where D is the number of independent variables. The Rosenbrock function (Figure 3.2) features regions of sharp gradients in select corners of the design space, which parabolically frame a relatively smooth valley that includes the global minimum.

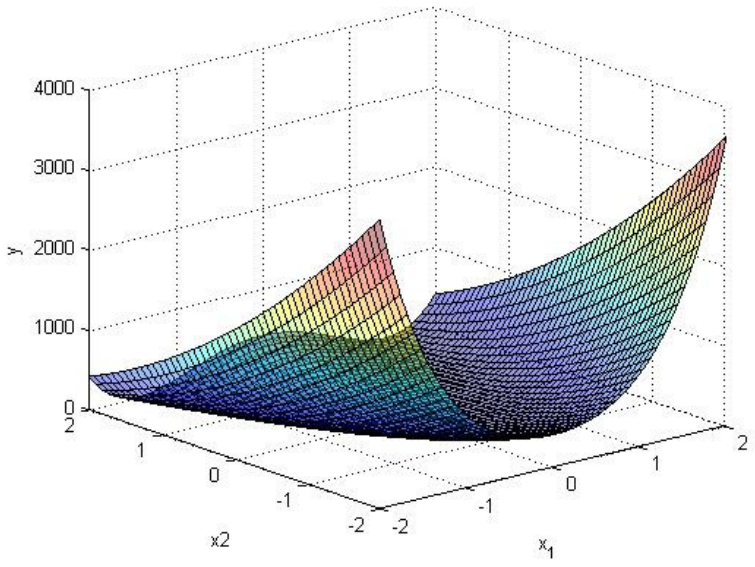


Figure 3.2: Two-dimensional Rosenbrock function.

The Rosenbrock function represents engineering problems with very steep gradients and rapid changes in response, which are not represented by the smoother

gradients of the kernel density functions. Example engineering applications include modeling the temperature profile on the surface of a welding specimen, where temperatures may increase very rapidly near a moving welding heat source (*cf.* [48]).

3.2.3 Modified Rosenbrock Function

A modified version of the Rosenbrock function is also used to test the performance of the three methods under consideration. In the modified version (Equation 3.15), the limits on the independent variable are reduced and the numerical coefficient is changed from 100 to 1, as follows:

$$f(\mathbf{x}) = \sum_{n=1}^{D-1} \left[\left(x_{n+1} - x_n^2 \right)^2 + \left(x_n - 1 \right)^2 \right], \qquad -1 \le x_n \le 1 \tag{3.15}$$

The modifications eliminate the sharp gradients and large changes in magnitude of the original Rosenbrock function. As illustrated in Figure 3.3, the modified version of the Rosenbrock function offers relatively smooth behavior, without the waving multimodality of the kernel density functions. The modified Rosenbrock function represents a wide variety of engineering application problems with responses that vary gradually, but not necessarily linearly, with changes in the input parameters.

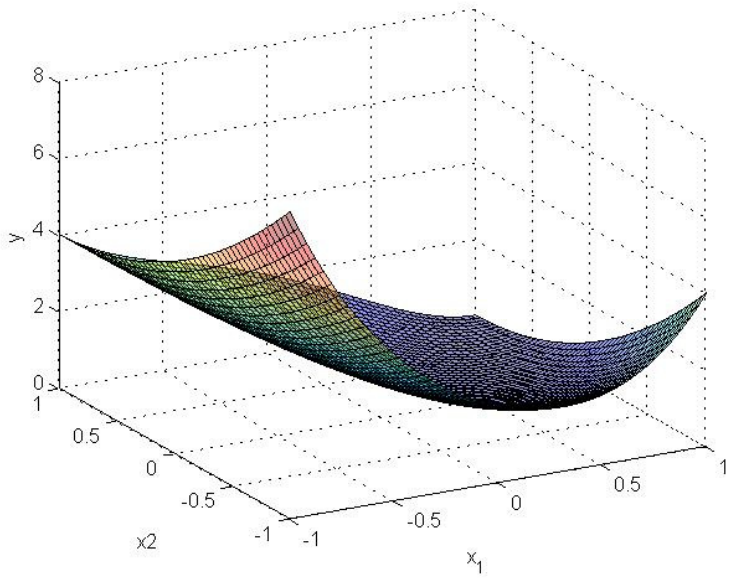


Figure 3.3: Two-dimensional modified Rosenbrock function.

3.3 EXPERIMENTAL DESIGN

3.3.1 Training and Test Point Sampling

The method for sampling training points from the base model can have a significant effect on the accuracy of the resulting metamodel. In contrast to physical experiments, which are stochastic in nature, deterministic computer models are not subjected to repeated sampling because their predictions typically do not vary unless randomness is built into the simulation model. Therefore, sampling strategies for computer experiments aim to fill the design space as thoroughly as possible [87]. There are several so-called space filling designs, including Latin hypercube designs [88], Hammersley sequence sampling [89], orthogonal arrays [90], and uniform designs [91].

Simpson *et al.* [77] find that uniform designs and Hammersley sequence sampling tend to fill the design space more evenly and provide more accurate results than Latin hypercube designs and orthogonal arrays. They also show that Hammersley sequencing

is preferred to uniform designs when large sets of training data can be afforded. In contrast to an expensive computer simulation, all of the test functions used in this paper can be sampled rapidly and large sets of training data are available. Therefore, Hammersley sequence sampling is selected as the method for generating training and test points in this study.

3.3.2 Performance Assessment

In this study, the metamodeling techniques are evaluated based on the number of training points required to achieve a predetermined error metric. The error metric used is the relative average absolute error (RAAE):

$$RAAE = \frac{\sum_{i=1}^{n} |y_i - \hat{y}_i|}{n\sigma} \tag{3.16}$$

where y_i is the actual value of the base model at the i_{th} test points, $\hat{y}_i$ is the predicted value from the metamodel, n is the number of sample points, and σ is the standard deviation of the response.

To determine the necessary number of training points, the training point sample size is increased incrementally (in increments as small as one additional training point) until an *RAAE* value of 0.25 is achieved. The locations of the training points are determined using Hammersley sequence sampling (HSS) [89]. Each time the sample size is increased, a new HSS is generated and the metamodel is completely refit for the new set of training points. The *RAAE* is calculated using one hundred test points per variable (100**D*), which are also generated with a Hammersley sequence.

Each metamodeling technique includes one or more parameters that can be tuned to improve the accuracy of the metamodel for a specific set of training data. In Equations (3.5) and (3.9), the Gaussian basis and kernel functions contain tuning parameters k and g

for radial basis functions and support vector regression, respectively. The value of these parameters has a significant effect on the quality of the resulting metamodel fit. In this study, these parameters are tuned manually by incrementally increasing or decreasing them to reduce the number of training points required to achieve the required RAAE, until a minimum number of training points is achieved. Contrary to RBF and SVR, kriging has correlation parameters θ_i that are identified automatically during the fitting process.

In addition to the number of sample points necessary to achieve the pre-specified error metric, the relative maximum absolute error (RMAE) is also used to compare the performance of the metamodeling techniques:

$$RMAE = \frac{\max\left|y_i - \hat{y}_i\right|}{\sigma}, \quad i = 1,...,n \tag{3.17}$$

Contrary to the RAAE in this study, the RMAE is not used as a threshold to determine the minimum required number of training points. It is only used to provide insights into how consistently each metamodel predicts the base function and, specifically, whether there are large deviations between the metamodel and its underlying base model in one or more regions of the design space.

The computational times required to build each model and to predict the 100*D test points are also recorded. All experiments are performed on a high performance computing Linux machine with two dual-core, 3.33GHz processors and 24 GB of physical memory.

Table 3.1: Complete experimental plan

Tests Performed				
Test	*Test Function*	*Scale*	*Test Points*	*Termination Criteria*
1	1 mode kernel	15 – 50 D	100*D	RAAE < 0.25
2	2 mode kernel	15 – 50 D	100*D	RAAE < 0.25
3	5 mode kernel	15 – 50 D	100*D	RAAE < 0.25
4	10 mode kernel	15 – 50 D	100*D	RAAE < 0.25
5	Rosenbrock	15 – 50 D	100*D	RAAE < 0.25
6	Modified Rosenbrock	15 – 50 D	100*D	RAAE < 0.25

The complete set of experiments (Table 3.1) tests kriging, radial basis functions, and support vector regression on KDE functions with 1, 2, 5, and 10 local maxima and the Rosenbrock functions with dimensionality ranging from 15 to 50 dimensions (in increments of 5). Each KDE experiment is repeated 5 times to account for the random placement of the initial kernels in the KDE, and the results are averaged. In total, each metamodeling method is subjected to 176 tests, resulting in 528 tests in all.

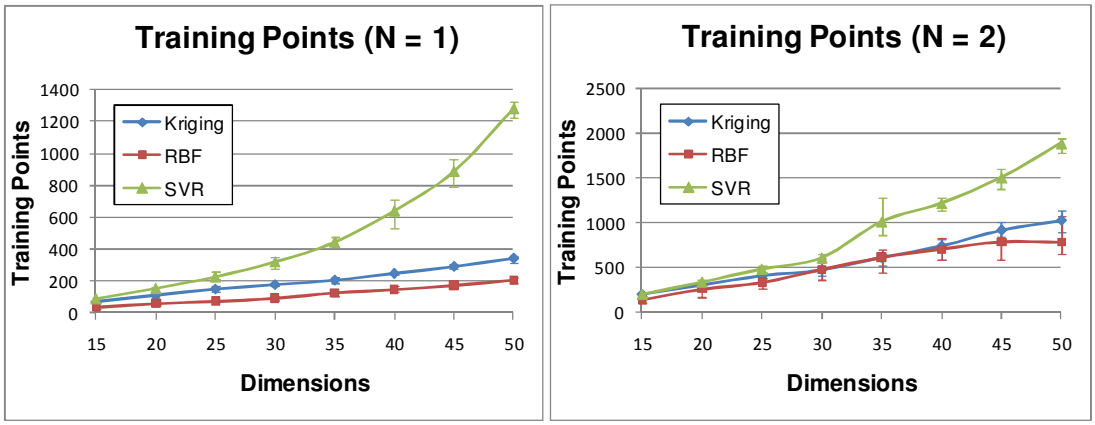
3.4 RESULTS AND DISCUSSION

3.4.1 Required Number of Training Points and Maximum Error

In Figure 3.4, the required number of training points for each test function is plotted versus dimensionality. For the KDE functions (Fig. 3.4a-d), the plotted points represent the number of training points required, averaged over the five experiments conducted with different random placements of the initial kernel in the KDE, and error bars indicate the largest and smallest number of training points required for each set of experiments. Regardless of the modality of the KDE function, the required number of training points increases steadily with the number of dimensions, as expected. In most cases, radial basis function and kriging metamodels require about the same number of training points, while support vector regression requires the largest set of training points of the three methods.

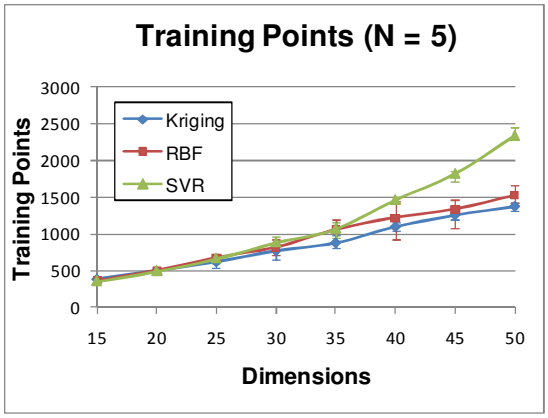
For the Rosenbrock and modified Rosenbrock functions (Figure 3.4e-f), the metamodels perform differently in comparison to each other. In both cases, kriging and SVR require similar numbers of training points, while RBF metamodels require the largest set of training points. The difference is most pronounced on the modified Rosenbrock function, where kriging and SVR reach an asymptote of fewer than 500 training points for this well-behaved function, regardless of dimension, while the number of training points required for RBF continues to increase with dimension. RBF's poor performance on the Rosenbrock functions is partially due to its underlying Gaussian basis function, which fits the KDE functions better than a Euclidean norm basis function, for example, but offers a poorer fit for the Rosenbrock functions.

In Figure 3.4a-d, it can be seen that the difference in the required number of training points between the metamodeling techniques diverges as the dimensionality of the KDE function increases. In 15 dimensions, the three methods require similar numbers of training points. However, as the number of independent variables increases, SVR requires increasingly more training points than kriging and RBF. The rate of the increase is smallest for RBF and kriging. These trends are consistent across all levels of modality, but they are most significant in the unimodal ($N = 1$) case.

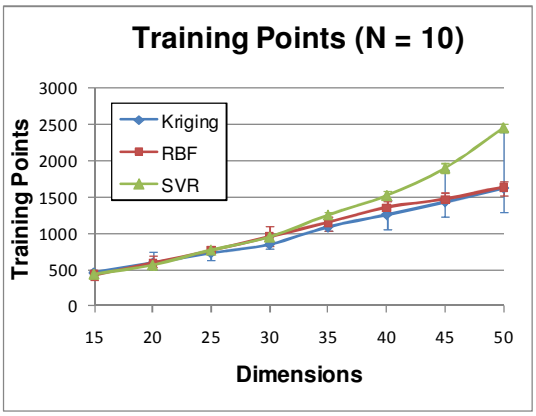


(a)

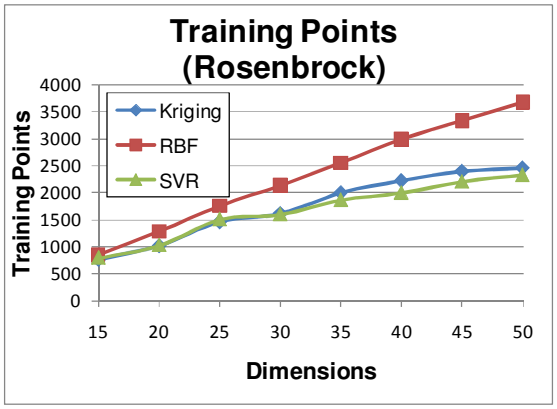
(b)



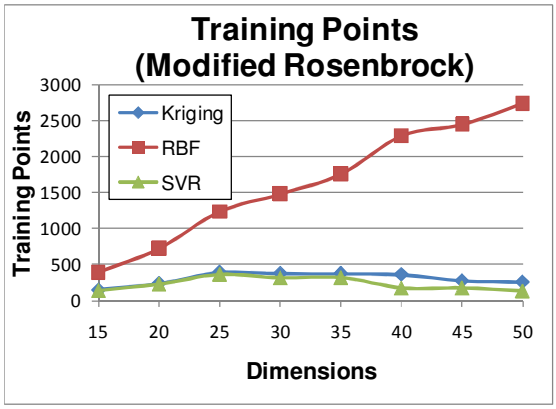
(c)



(d)



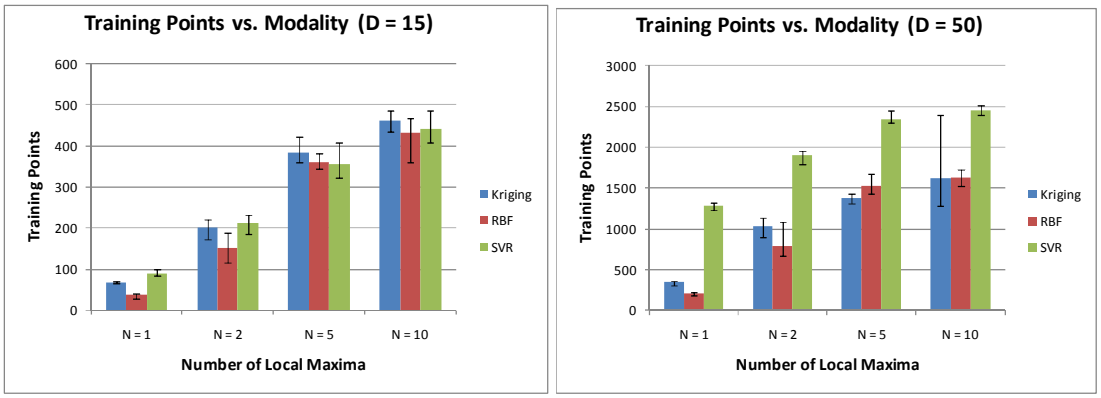
(e)



(f)

Figure 3.4: Required number of training points for 1, 2, 5, and 10 mode KDE (a-d), Rosenbrock function (e) and Modified Rosenbrock function (f). Error bars on Figures 3.4a-d indicate maximum and minimum values for 5 different KDE trials.

Careful observation of Figure 3.4a-d reveals that the number of required training points increases with the complexity (number of local maxima) of the KDE functions. In Figure 3.5, the required number of training points is plotted versus modality for specific dimensions. Results are shown for all three metamodeling techniques applied to the 15 and 50 dimensional problems (Fig. 6a and 6b, respectively) with 1, 2, 5, and 10 local maxima. When the number of dimensions is low (D = 15), the number of required training points appears to increase linearly with modality for all three methods. However, when the number of dimensions is high (D = 50), the required number of training points appears to approach an asymptote. This trend is explained by the fact that all of the metamodeling techniques have difficulty capturing the modality of the KDE test functions in high dimensions.



(a) (b)

Figure 3.5: Number of training points versus modality for D = 15 (a) and D = 50 (b). Error bars indicate maximum and minimum numbers of training points for 5 different KDE trials.

Since five experiments were conducted for each combination of KDE dimensionality and modality, the results support some insights into the sensitivity of the various metamodeling techniques to the spatial distribution of local maxima within the design space. In general, higher sensitivity, represented by larger error bars in Figures

3.4a-d, indicates an interaction effect between the number of required data points and the distribution of local maxima. For highly multimodal, high-dimensional problems, kriging tends to have the highest sensitivity with large error bars in Figure 3.4d for 45 and 50 dimensions. For other combinations of modality and dimensionality, the sensitivity of RBF and SVR is equivalent and sometimes greater.

The average absolute error metric was used to determine the required number of training points. It is a good indicator of the global fit of the metamodel, but it is also important to consider the maximum error among all test points, because designers are often most interested in the response at a single point or at a small sub-region of the design space. In Figure 3.6, the relative maximum absolute error (RMAE) is plotted for each test case. The number of training points used in these cases is equal to the required number to achieve an RAAE of 0.25.

For the KDE functions (Figure 3.6a-d), the RMAE generally increases with the modality of the problem. In the unimodal case, the RMAE remains mostly constant for all three methods with the exception of SVR, for which the RMAE increases in the 45 and 50 dimensional problem. In the 2, 5, and 10 mode cases, the RMAE increases slightly for all methods, relative to the unimodal case, with RBF exhibiting the highest RMAE in most cases. These higher levels of RMAE for the multimodal cases reflect the metamodeling techniques' difficulty in predicting the local maxima for the multimodal functions.

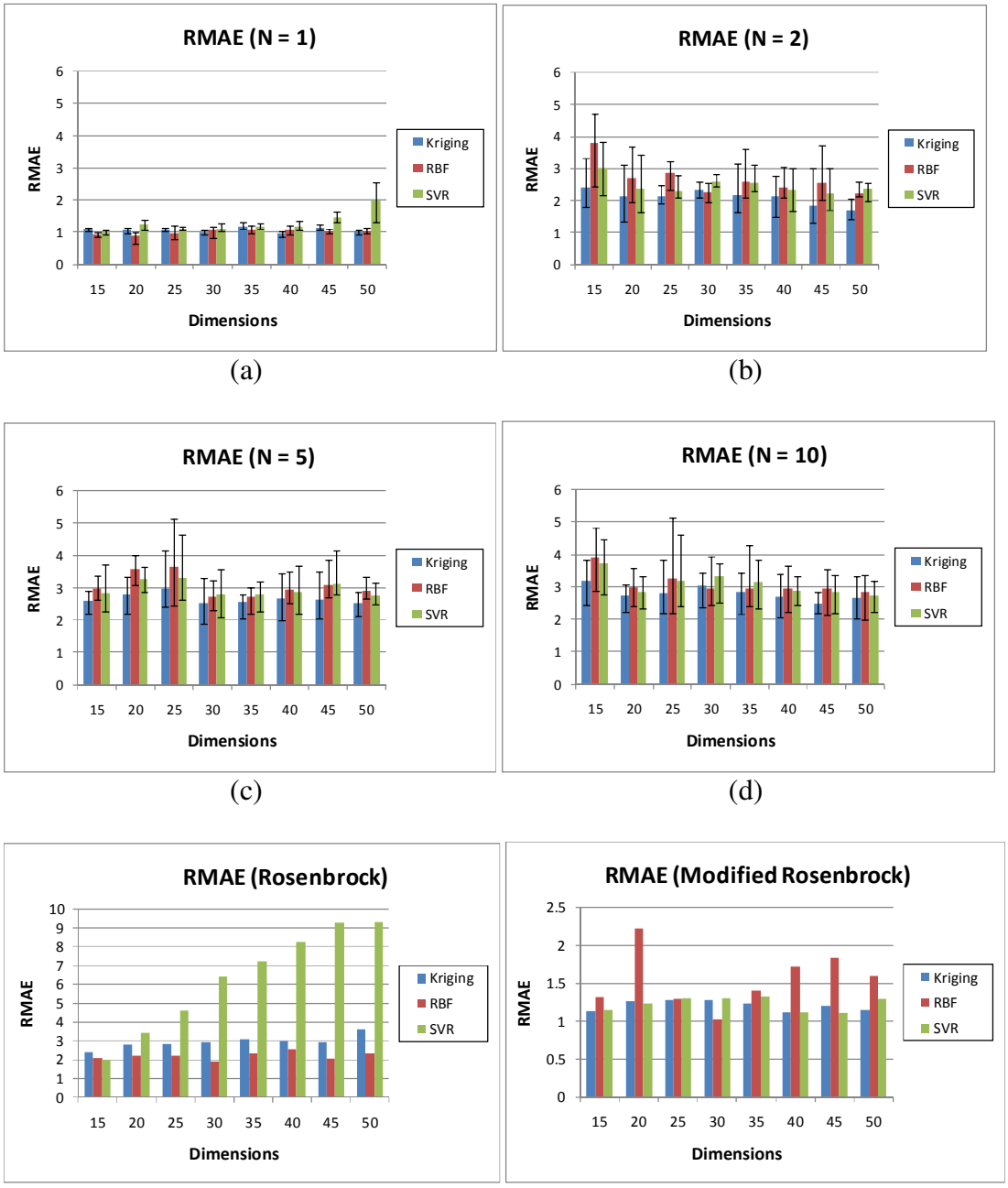


Figure 3.6a-f: RMAE for 1, 2, 5, and 10 mode KDE (a-d), Rosenbrock (e) and Modified Rosenbrock (f) functions. Error bars on Figures 3.6a-d indicate maximum and minimum RMAE values for 5 different KDE trials.

The RMAE for the Rosenbrock function is presented in Figure 3.6e. In this figure, the RMAE of kriging and RBF is low and stays constant as the dimensionality of the problem increases. However, the RMAE of SVR is significantly higher than that of the other functions, especially in high dimensions. The Rosenbrock function features sharp increases in the response at select corners of the design space, and the results reflect SVR's difficulty in predicting the response in those regions.

The RMAE values of the modified Rosenbrock (Figure 3.6f) are much smaller in magnitude than those for the Rosenbrock function. Also, the values of the RMAE for the three metamodeling methods in this case remain relatively constant as the dimensionality of the function increases. These trends reflect the fact that all of the metamodeling techniques can model this relatively smooth nonlinear function with high levels of accuracy, although RBF requires very large numbers of training points to do so.

Based on the results of this study, kriging appears to be the method that is most capable of approximating a variety of functions in high dimensions. For the KDE functions, which represent problems with various levels of modality, RBF performs similarly to kriging. SVR requires the largest number of training points for the high dimensional KDE functions, but this gap in performance narrows as the modality of the problem increases. RBF requires the largest number of training points for the high-dimensional Rosenbrock functions, which represent problems with sharp gradients alongside regions of very smooth gradients. Kriging and SVR perform similarly with respect to the number of required training points for the Rosenbrock functions, but SVR exhibits higher RMAE on the original Rosenbrock function.

3.4.2 Training Time

Training time (Figure 3.7) is measured in real time seconds using the built in stopwatch function in MATLAB®. When presenting these results, it is important to consider the fact that each metamodeling technique is trained with a different number of training points, since computational expense tends to increase with the number of training points. Therefore, the total training time is divided by the number of training points required by each model.

In Figure 3.7, the average training times are plotted versus dimensions for the 1, 2, 5, and 10 mode KDE and the Rosenbrock functions. In most cases, the training time per training point for RBF was significantly higher than that of the other two methods. The exception is in the unimodal KDE case, in which kriging had the slowest training time per training point. In all other cases, kriging had the second slowest training time per training point and support vector regression had the fastest training time per training point of all three methods.

In previous metamodeling studies, RBF's training time was found to be less than that of kriging on low-dimensional problems (*cf.* [44]). RBF is much slower than kriging in this study, primarily because it uses a pseudo-inverse operation to determine the weights in Equation 3.4. The pseudo-inverse operation is much slower than a standard inverse or matrix division operation in MATLAB®, but it is more general because it can operate on matrices that are nearly singular. The pseudo-inverse operation accounts for at least a 90% increase in training time, relative to standard MATLAB® matrix division operations.

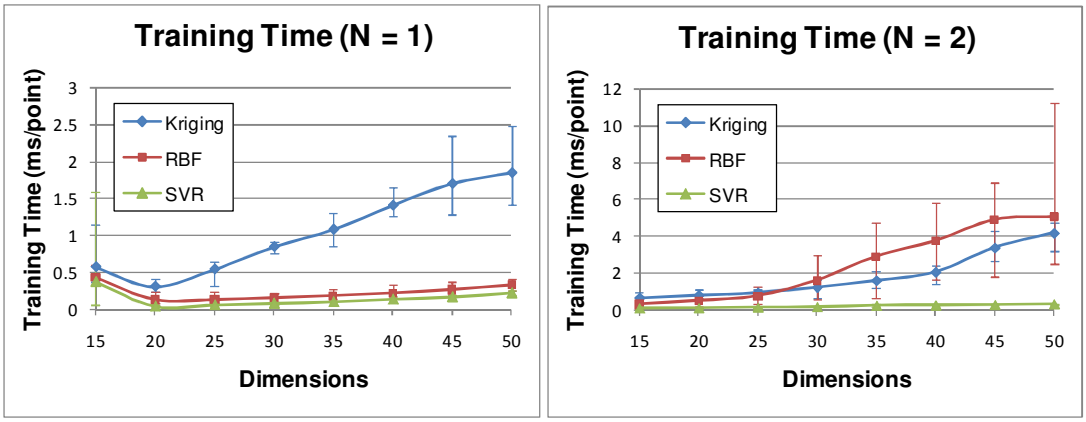


Figure 3.7a-f: Training time for 1, 2, 5, and 10 mode KDE (a-d), Rosenbrock (e) and Modified Rosenbrock (f) functions. Error bars on Figures 3.7a-d indicate maximum and minimum training times for 5 different KDE trials.

SVR could be expected to exhibit short training times, relative to kriging. Like RBF, the number of parameters to be fit scales with the number of training points, rather than the number of dimensions, as in kriging. Unlike kriging, which uses a direct search algorithm for fitting parameters (Lophaven *et al.* 2002), the SVR algorithm uses a relatively fast sequential minimal optimization (SMO) algorithm (Platt 1998) that scales between linearly and quadratically with the number of training points and guarantees convergence to a unique global optimum.

3.4.3 Prediction Time

In Figure 3.8, the time to predict 100**D* new points is plotted versus dimensions for 1, 2, 5, and 10 mode KDEs and the Rosenbrock functions. As with training time, the total prediction times were divided by the number of training points. This normalization was performed because models that are trained with more training points have more features and are therefore more mathematically intensive. For example, the number of basis functions in an RBF model is equal to the number of training points because the locations of the basis function centers are chosen to be coincident with the training points.

The prediction time per training point of the three methods differed significantly from the training time. In this case, kriging was always the slowest of the three methods. Radial basis functions and SVR had similar prediction times with SVR being only marginally faster than RBF. Also, the prediction times per training point were approximately uniform across all of the test functions. That is, the prediction time per training point depends on the number of dimensions but not the modality of the problem.

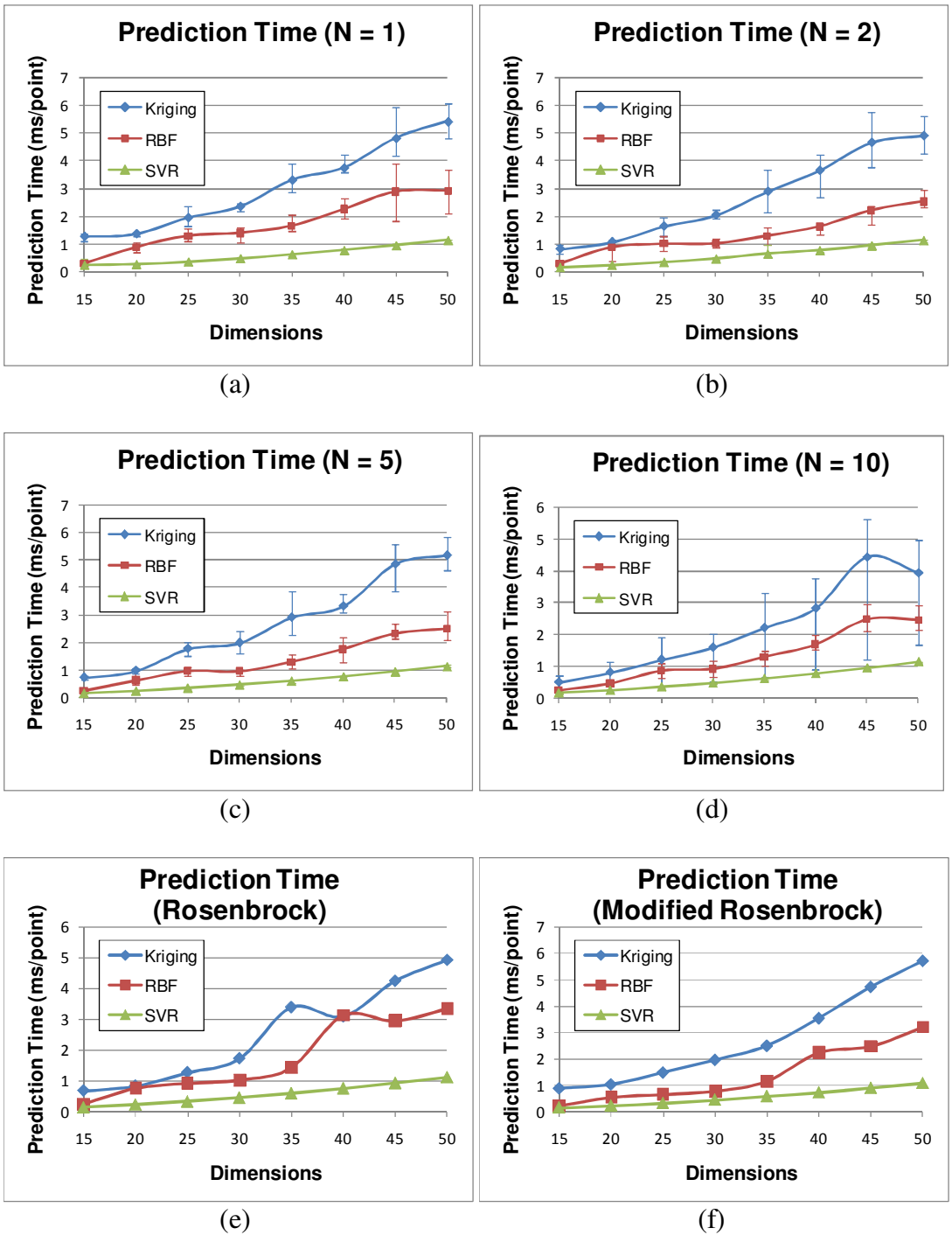


Figure 3.8a-f: Prediction time for 1, 2, 5 and 10 mode KDE (a-d), Rosenbrock (e), and Modified Rosenbrock (f) functions. Error bars on Figures 3.8a-d indicate maximum and minimum prediction times for 5 different KDE trials.

The relative ranking of metamodeling methods is usually the same, regardless of whether prediction time per training point or total training time is used as a comparison metric. One exception to this rule is in the unimodal KDE case in high dimensions (35 and up), in which the total prediction time of SVR is greater than that of RBF but less than that of kriging.

3.4.4 Discussion of Results

The results of these experiments provide insights into the scalability of each of the methods to higher dimensions. In practice, selection of the best metamodeling method depends on the situation. For example, a computationally expensive base model imposes practical limitations on the number of available training points.

In high-dimensional multimodal problems for which few training points are available, RBF and kriging are advantageous because they require the fewest training points to achieve an accurate fit. SVR requires a larger number of training points for these types of problems. These results differ from some of the low-dimensional studies in the literature. For example, Jin *et al.* [44] show that RBF is superior to kriging in terms of average accuracy but that kriging and RBF have similar accuracies when the number of independent variables is small. Kim *et al.* [47] show that kriging is more accurate than SVR and RBF for highly nonlinear problems with less than 10 variables. The results in this paper indicate that kriging is competitive with RBF in terms of average and maximum error for high-dimensional, multimodal problems. However, for the highly nonlinear Rosenbrock function, kriging is competitive with SVR but significantly outperforms RBF in terms of average error, as indicated by the required number of training points. Although kriging performs most consistently in this study, it should be noted that there is flexibility in how an RBF metamodel is constructed. For example,

Shan and Wang [92,93] proposed a method for constructing high-dimensional metamodels that combines RBF with a high dimensional model representation (HDMR). Shan and Wang demonstrate that the hybrid RBF-HDMR technique is more accurate and efficient (in terms of the number of required data points) than competing metamodeling techniques. Accordingly, the RBF-HDMR technique is likely to require fewer data points than the RBF technique tested in this study.

Previous studies have shown that kriging has a slow build time compared to RBF in cases where the number of independent variables is low [44]. The present study shows that RBF has a very slow build time compared to kriging in high dimensions, particularly when the function is highly multimodal and requires a large number of training points. However, as discussed in Section 3.4.1, this trend is primarily due to the MATLAB® implementation of RBF for this study, which uses more general but slower matrix inversion techniques to fit the RBF metamodel. The results for SVR presented here are consistent with previous studies that show that SVR has a very fast build time, relative to kriging [45].

3.5 CONCLUSIONS

In this chapter, alternative metamodeling techniques are evaluated with respect to their ability to approximate base functions with large numbers of independent variables. In general, kriging appears to be the dominant method because of its ability to approximate all of the test functions in this study accurately with fewer or equivalent numbers of training points, relative to SVR and RBF. SVR requires more training points to achieve the same level of average accuracy when applied to the KDE functions, while RBF requires more training points when applied to the Rosenbrock and modified Rosenbrock functions. RBF metamodels are found to have extremely slow build times

when the number of training points is large. SVR consistently has the fastest build and prediction times. Another advantage of kriging is its ease of use because the correlation function parameters are automatically tuned during the training process. This is not the case with SVR and RBF because they both require user-defined inputs that have a significant effect on the quality of the resulting fit.

Future work could extend the insights of this study in several ways. For example, it would be interesting to develop a set of high-dimensional engineering application problems for comparing alternative metamodeling techniques. In this work, mathematical functions were used, primarily because the number of dimensions could be scaled independently of the underlying features of the problem, but engineering application problems are likely to provide physical insights into the relative strengths and weaknesses of the methods that could prove useful to practitioners. Also, in this study, the metamodeling methods were compared by increasing the number of training points until a threshold RAAE value was achieved, and RMAE was calculated subsequently. It would be interesting to replace RAAE with RMAE or to simply specify a fixed number of training points for each technique and calculate and compare various metrics such as RAAE, RMAE, and R-squared. Such a study could give the practitioner additional insights into how the various metamodeling methods handle sparse training data. It would also be interesting to investigate different versions of the metamodeling techniques, such as the use of different types of kernel functions in SVR or basis functions in RBF. Finally, the large number of training points required for standard versions of RBF, kriging, and SVR to adequately fit high-dimensional problems highlights the need for more efficient high-dimensional metamodeling techniques. A recent example is the integration of RBF with high-dimensional model representation (HDMR) to form the RBF-HDMR metamodeling technique [92,93]. Although the

present study focuses on comparing the high-dimensional performance of relatively standard versions of each metamodeling technique, it would be interesting to compare the performance of these standard versions to other versions, such as the RBF-HDMR, that have been updated for improved high-dimensional performance.

Chapter 4: The Classifier-Guided Sampling Method

Metamodeling is a well-established method for providing fast and accurate approximations of expensive computer models that have continuous variables and smooth, continuous responses. However, many engineering problems have discrete variables and discontinuous underlying functions. In this chapter, the classifier-guided sampling (CGS) method for solving design optimization problems with discrete variables and fully discontinuous objective functions is presented. The CGS method is specifically designed to solve problems in which the response cannot be divided into continuous sub-regions and cannot be interpolated to estimate performance between training points. The CGS method uses a Bayesian network classifier in place of a metamodel to approximate total system behavior. A classifier is similar to a metamodel in that it is used to predict future instances based on a set of known observations or training points. However, a classifier assigns categorical class labels to unevaluated test points rather than attempting to predict the resulting response in continuous space. The CGS method exploits this property of classifiers by using it to guide a sampling process towards combinations of continuous and/or discrete design variable values with high probabilities of yielding preferred performance. In the next section, Bayesian network classifiers are discussed. In Section 4.2, the details of the CGS method are outlined. Section 4.3 presents the results of tests in which the performance of the CGS method is compared to genetic algorithms and random search when applied to a set of three discrete variable optimization problems. In Section 4.4, the time complexity if the CGS algorithm is investigated.

4.1 BAYESIAN NETWORK CLASSIFIERS

In machine learning, a classifier is used to assign categorical class labels to test points that have known feature attributes but unknown class labels [94]. The classifier is trained using a set of feature vector / class label pairs that are generally obtained experimentally. There are numerous methods available for classification including decision trees [95], learned rules [96], neural networks [97], Bayesian network classifiers [98], and support vector machines [99]. In this section, a classifier that uses Bayesian network (BN) [100] derived probability distributions is presented. The Bayesian network classifier is selected for use in the CGS method because it is a probabilistic method that provides the user with a probability that each test point belongs to a class, while many other classification methods only provide the class label as an output. This property is critical to the sampling step of the CGS method that is described in Section 4.2.

4.1.1 Bayesian Networks

A Bayesian network (BN) is a probabilistic graphical model [101] that is used to represent conditional dependencies between a set of random variables. This task is achieved with the use of a directed acyclic graph (DAG). A DAG consists of a set of nodes that are connected by directed edges. The directed edges must be configured in such a way that the graph is acyclic, *i.e.* it is impossible to start at a particular node and follow a path of edges that loops (cycles) back to that node. In a BN, the nodes of the DAG represent random variables, and the directed edges are used to represent conditional dependencies.

Bayesian networks provide a convenient and intuitive way to represent conditional dependencies among random variables. The joint probability P of a D-dimensional vector of random variables $\mathbf{x}$ is written as

$$P(\mathbf{x}) = P(x_1, x_2, ..., x_D) \tag{4.1}$$

By repeatedly applying the chain rule for conditional probabilities, $P(\mathbf{x})$ can be expanded as a product of conditional probabilities as follows:

$$P(\mathbf{x}) = P(x_D \mid x_{D-1}, ..., x_1) P(x_{D-1} \mid x_{D-2}, ..., x_1) ... P(x_2 \mid x_1) P(x_1) \tag{4.2}$$

Each conditional probability on the right-hand side of Equation (4.2) is represented by a node in a BN graph. For example, consider the case in which there are four random variables. One possible BN for these four variables is shown in Figure 4.1:

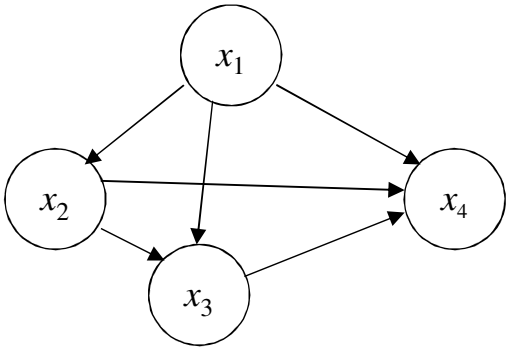


Figure 4.1: Fully dependent Bayesian network graph

The graph in Figure 4.1 represents a special case which is referred to as a "fully dependent" or "fully connected" BN. The joint probability $P(\mathbf{x})$ that corresponds to the graph in Figure 4.1 is given by

$$P(\mathbf{x}) = P(x_4 \mid x_3, x_2, x_1) P(x_3 \mid x_2, x_1) P(x_2 \mid x_1) P(x_1) \tag{4.3}$$

In Equation 4.3, the conditional probability function for each variable's node depends directly on the nodes that are connected to that particular variable. For the purpose of convenience, variables are typically labeled so that the edges are directed in "topological order", meaning that the edges are to be directed so that the ancestors of a particular node (parents, parents of parents, *etc.*) are always nodes that correspond to

lower numbered variables and descendants of a particular node are always nodes that correspond to higher numbered variables.

Nodes that are not connected indicate conditional independence between their respective variables. To represent the general case, the joint probability $P(\mathbf{x})$ can also be written as

$$P(\mathbf{x}) = \prod_{i=1}^{D} P_i(x_i \mid \mathbf{pa}_i) \tag{4.4}$$

where $\mathbf{pa}_i$ are the parents of the node corresponding to the random variable x_i. For example, consider the BN shown in Figure 4.2:

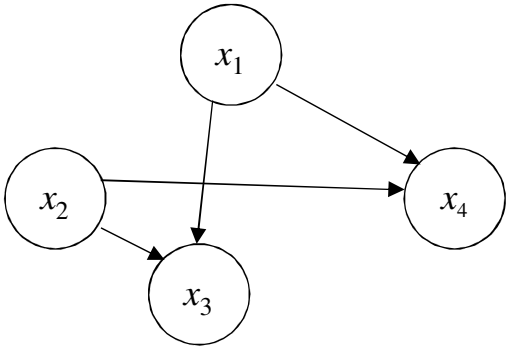


Figure 4.2: Partially dependent Bayesian network

In Figure 4.2, x_3 and x_4 both have parents x_1 and x_2, and x_1 and x_2 have no parent nodes. Therefore, the joint probability $P(\mathbf{x})$ for the BN shown in Figure 4.2 is given by Equation (4.5):

$$P(\mathbf{x}) = P(x_4 \mid x_2, x_1) P(x_3 \mid x_2, x_1) P(x_2) P(x_1) \tag{4.5}$$

It is not uncommon for there to be no directed edges in a BN. This situation (Figure 4.3) occurs when there are no conditional dependencies among variables, and is another special case referred to as the "fully independent" BN.

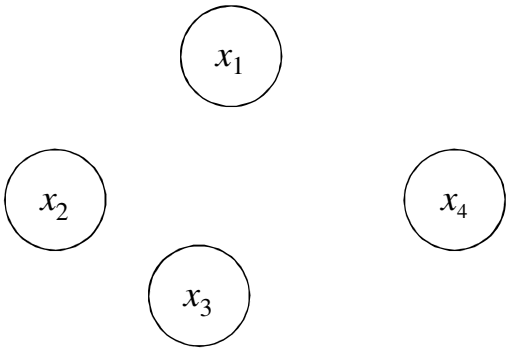


Figure 4.3: Fully independent Bayesian network

The joint probability $P(\mathbf{x})$ for the BN shown in Figure 4.3 is given by Equation (4.6):

$$P(\mathbf{x}) = P(x_4)P(x_3)P(x_2)P(x_1) \tag{4.6}$$

In this section, the concept of a BN was introduced. The joint probability functions that result from BNs are a critical component of Bayesian classifiers, which are described next.

4.1.2 Bayesian Classifiers

Consider a K category classification problem, where c_k represents class k and $k = [1,2,\ldots,K]$. Assume the classification is to be performed for a D-dimensional design point $\mathbf{x}$ with unknown classification. Bayesian networks and Bayes formula can be used to estimate the posterior probability of class c_k given the design point $\mathbf{x}$, $P(c_k|\mathbf{x})$, according to

$$P(c_k \mid \mathbf{x}) = \frac{P(\mathbf{x} \mid c_k)P(c_k)}{P(\mathbf{x})} = \frac{P(\mathbf{x} \mid c_k)P(c_k)}{\sum_{k=1}^{K} P(\mathbf{x} \mid c_k)P(c_k)} \tag{4.7}$$

where $P(\mathbf{x}|c_k)$ is the class conditional probability of the test point $\mathbf{x}$ and $P(c_k)$ is the prior probability of class k. Design $\mathbf{x}$ is classified as a member of the class c_k that has the highest $P(c_k|\mathbf{x})$ when compared to all other $P(c_i|\mathbf{x})$ for $i \in [1,\ldots,K]$. There are two key

parameters of the classifier: the prior probability, $P(c_k)$, and the class conditional probability, $P(\mathbf{x}|c_k)$.

In general, the user may define the prior probabilities of each class using any preferred method. For example, if there is no reason to believe that one class is more probable than any other, each $P(c_k)$ can be set equal for all k. In this research, the prior probabilities, $P(c_k)$, are estimated using the frequency of occurrence of each class in the training set according to

$$P(c_k) = \frac{N_k + 1}{N + K} \tag{4.8}$$

where N is the total number of training points, N_k is the total number of training points for class k, and K is the total number of classes. One count is added to the numerator in Equation (4.8) to avoid probabilities of zero when in fact it is known that each class has prior probability that is at least greater than zero. K counts are added to the denominator to normalize the distribution.

The class conditional probabilities, $P(\mathbf{x}|c_k)$, are joint probability distributions with design variable and class dependence relationships that can be represented by BN graphs. In a classification problem, each design variable is always conditionally dependent on the class, but they may or may not be conditionally dependent on the other variables. Therefore, the chain rule can be used to expand the class conditional probability as follows:

$$P(\mathbf{x} \mid c_k) = P\left(x_D \mid \mathbf{pa}_D^x, c_k\right) P\left(x_{D-1} \mid \mathbf{pa}_{D-1}^x, c_k\right) \ldots P\left(x_2 \mid \mathbf{pa}_2^x, c_k\right) P\left(x_1 \mid c_k\right) \tag{4.9}$$

where $\mathbf{pa}_i^x$ are the parent *design variable* nodes of the design variable x_i. Each term in the product on the right-hand side of Equation (4.9) is conditionally dependent on the

class c_k, but conditional dependencies on other variables are defined by the connectivity of the BN graph.

There are two well-studied special cases of network connectivity for classification. In the first special case, known as the "fully dependent" case, the BN is fully connected. A fully connected BN graph for a classifier with three design variables is shown in Figure 4.4.

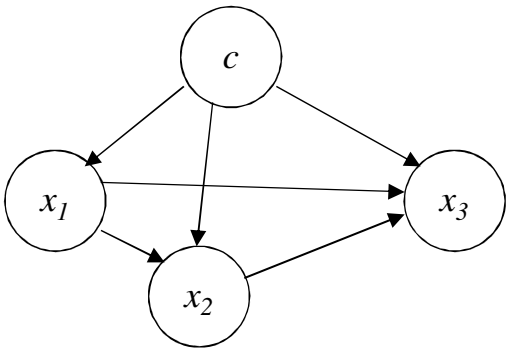


Figure 4.4: Fully dependent Bayesian network classifier

The class conditional probability for the BN shown in Figure 4.4 is given by Equation (4.10):

$$P(\mathbf{x} \mid c) = P(x_3 \mid x_2, x_1, c)\, P(x_2 \mid x_1, c)\, P(x_1 \mid c) \tag{4.10}$$

A drawback of the fully dependent case is that estimation of the class conditional probabilities is not practical in high-dimensional space when a limited number of training points is available. The problem is greatly simplified by assuming the design variables are independent given the class. This assumption results in a second special case, known as the Naïve Bayes classifier (Figure 4.5).

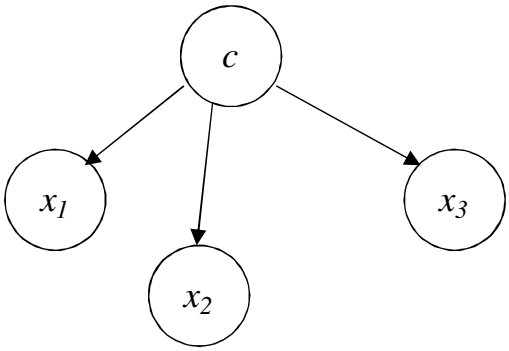


Figure 4.5: Naïve Bayes classifier

By assuming independence among all design variables, the class conditional probability of the BN in Figure 4.5 is given by

$$P(\mathbf{x}\,|\,c) = P(x_3\,|\,c)\,P(x_2\,|\,c)\,P(x_1\,|\,c) \tag{4.11}$$

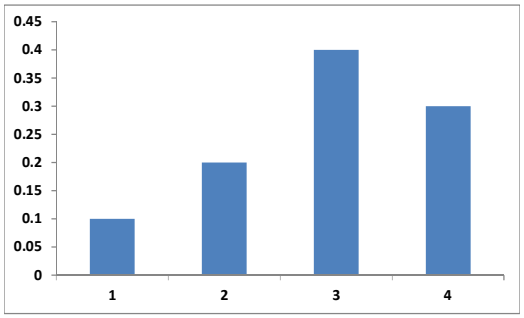
In general, any probability distribution can be used to estimate the values of each $P(\mathbf{x}|c_k)$ in Equation (4.7). In this research, multinomial distributions are the most appropriate choice because the focus here is specifically on discrete variable classification problems. A multinomial distribution is a frequency-based distribution, and there are D distributions for each class, where D is the number of design variables. Each multinomial distribution is a normalized histogram in which there is a bin for each discrete value that each variable is permitted to take. For example, consider a set of 100 training points with the two classes 'true' and 'false', and one variable, x_1. Say x_1 can assume integer values from 1 to 5. If there are 50 training points that are labeled as class 'true', and for ten of those $x_1 = 1$, $P(x_1=1|true)$ will be equal to (10/50) or 0.2.

To visualize these distributions in the context of Bayesian networks, consider the two-dimensional set of training points in Table 4.1. In this training data set, there are two variables, x_1 and x_2, that can assume integer values 1, 2, 3, or 4. For simplicity, assume all of the ten training points are assigned the class label c_1.

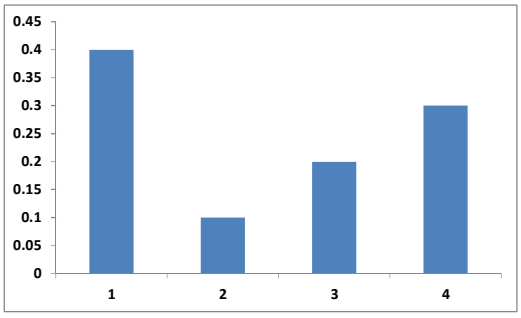
Table 4.1: Example training data set

x_1	4	2	4	3	1	3	3	4	2	3
x_2	2	1	1	4	4	1	3	4	3	1
Class	c_1	c_1	c_1	c_1	c_1	c_1	c_1	c_1	c_1	c_1

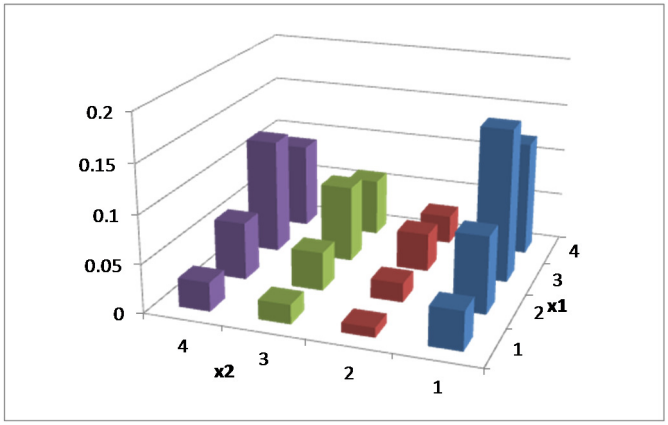
This training set will be used to generate multinomial distributions for both a Naïve Bayes and a fully dependent BN classifier. The class conditional probabilities for the Naïve Bayes and fully connected classifier are shown in Figures 4.6 and 4.7, respectively.



(a) $P(x_1 \mid c_1)$

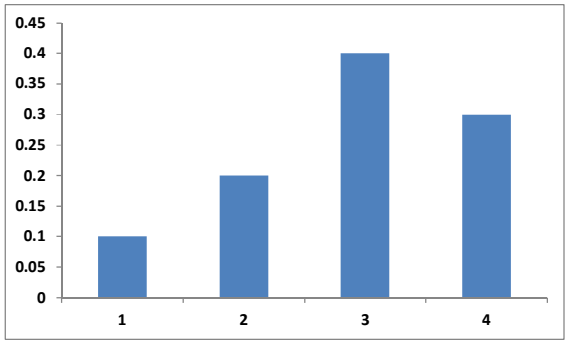


(b) $P(x_2 \mid c_1)$

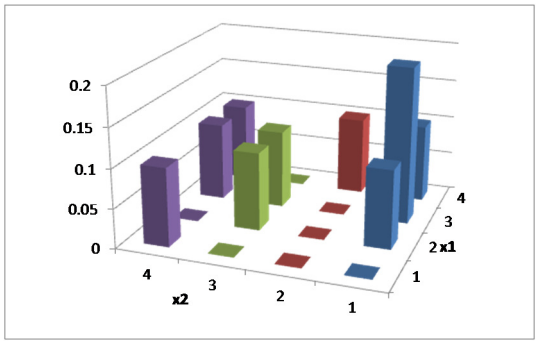


(c) $P(\mathbf{x} \mid c_1) = P(x_2 \mid c_1) P(x_1 \mid c_1)$

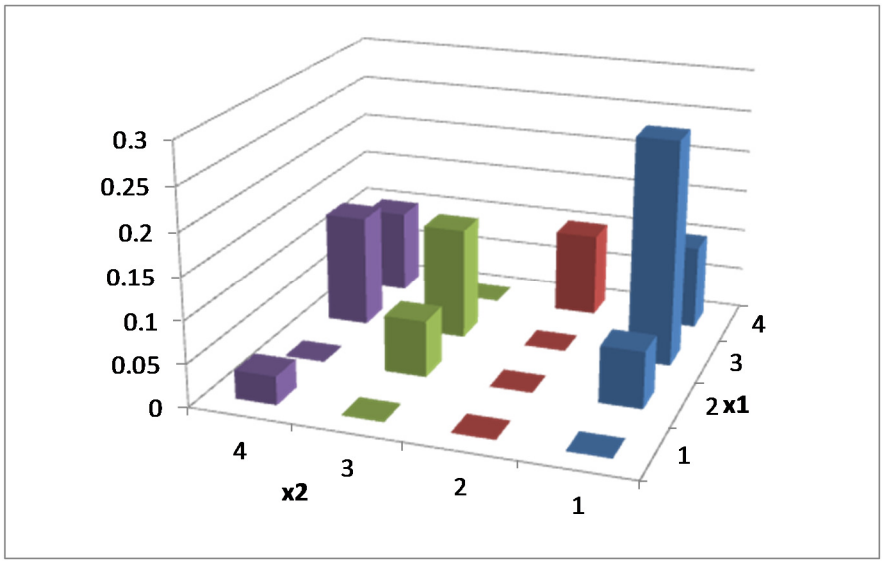
Figure 4.6: Naïve Bayes multinomial distributions for x_1 (a), x_2 (b), and the complete class conditional probability distribution for **x** (c).



(a) $P(x_1 \mid c_1)$



(b) $P(x_2 \mid x_1, c_1)$



(c) $P(\mathbf{x} \mid c_1) = P(x_2 \mid x_1, c_1) P(x_1 \mid c_1)$

Figure 4.7: Fully connected multinomial distributions for x_1 (a), x_2 (b), and the complete class conditional probability distribution for **x** (c).

In the Naïve Bayes case, $P(\mathbf{x}|c_1)$ is formed from the product of two independent distributions, each of which is one-dimensional. However, in the fully connected case the variables are not assumed to be independent of each other. Therefore, a two dimensional distribution, $P(x_2|x_1,c_1)$, is required that needs significantly more training points to fill adequately. For the 10 training points in this example, there are a significant number of empty bins in the distribution. Counts of zero in the multinomial distribution result in a probability estimate of zero when the classifier is used to predict any variable

combinations that did not appear in the training set. As the number of parent nodes for a particular variable grows, this problem becomes more severe as the design space and the number of admissible discrete variable combinations grows exponentially. This problem can be alleviated with the use of a pseudocount, *i.e.* adding one count to each bin so that no variable combination has a probability of zero. Still, this remedy must be used cautiously so as not to unfairly bias the classifier towards unwanted inaccuracies. Therefore, there are advantages and disadvantages to increasing the number of conditional dependencies in a BN classifier. Using the simplest (least conditionally dependent) BN structure possible enables the class conditional probability distributions to be adequately populated with fewer training points. On the other hand, a BN that models more conditional dependencies between nodes better captures the relationships between design variables and may therefore improve classifier-guided search algorithm performance. The disadvantage of more complex BN classifiers is that significantly more training points may be required for the classifier to provide meaningful classifications.

BN classifiers have several important properties that make them ideal for use in the classifier-guided sampling method. First, the user can easily control the conditional dependence of the design variables in the classifier, making it easy to use designer intuition to tailor the classifier for various problem-specific cases. Second, BN classifiers provide the user with a probability that a test point belongs to a certain class, in addition to a simple categorical class label. This property is critical to the sampling step in the classifier-guided sampling method, which is developed in the next section.

4.2 THE CLASSIFIER-GUIDED SAMPLING METHOD

The classifier-guided sampling (CGS) method uses a classifier in place of a metamodel to provide predictions of total design space performance. The classifier prediction is based on a set of training points from the expensive base model. A classifier differs from a metamodel in that it cannot give quantitative predictions on a continuous scale. A classifier can, however, provide a qualitative estimate of an objective function value by pairing each candidate solution with a categorical class label. A classifier is used in the CGS method to asses a large set of candidate solutions quickly without requiring an expensive simulation for each point. The classifier outputs are used to guide the sampling process towards optimal or near optimal solutions. This method is especially useful for cases in which the concept of distance between points is irrelevant or undefined.

The CGS method (Figure 4.8) begins by executing expensive simulations for a set of randomly generated training points. The outputs of the expensive base simulation are assigned qualitative class labels (*e.g.* 'low' / 'high' quality) based on their objective function values. This task is achieved by defining a class threshold, T_C, which serves as a decision boundary for assigning class labels to training points based on their objective function values. If the design goal is to maximize an objective function and the training point has an objective function value higher than T_C, it is given a class label of 'high' quality. A training point is given a class label of 'low' quality if it is less than T_C. The rule is reversed for minimization problems in which case a training point is given a class label of 'high' if it is less than T_C.

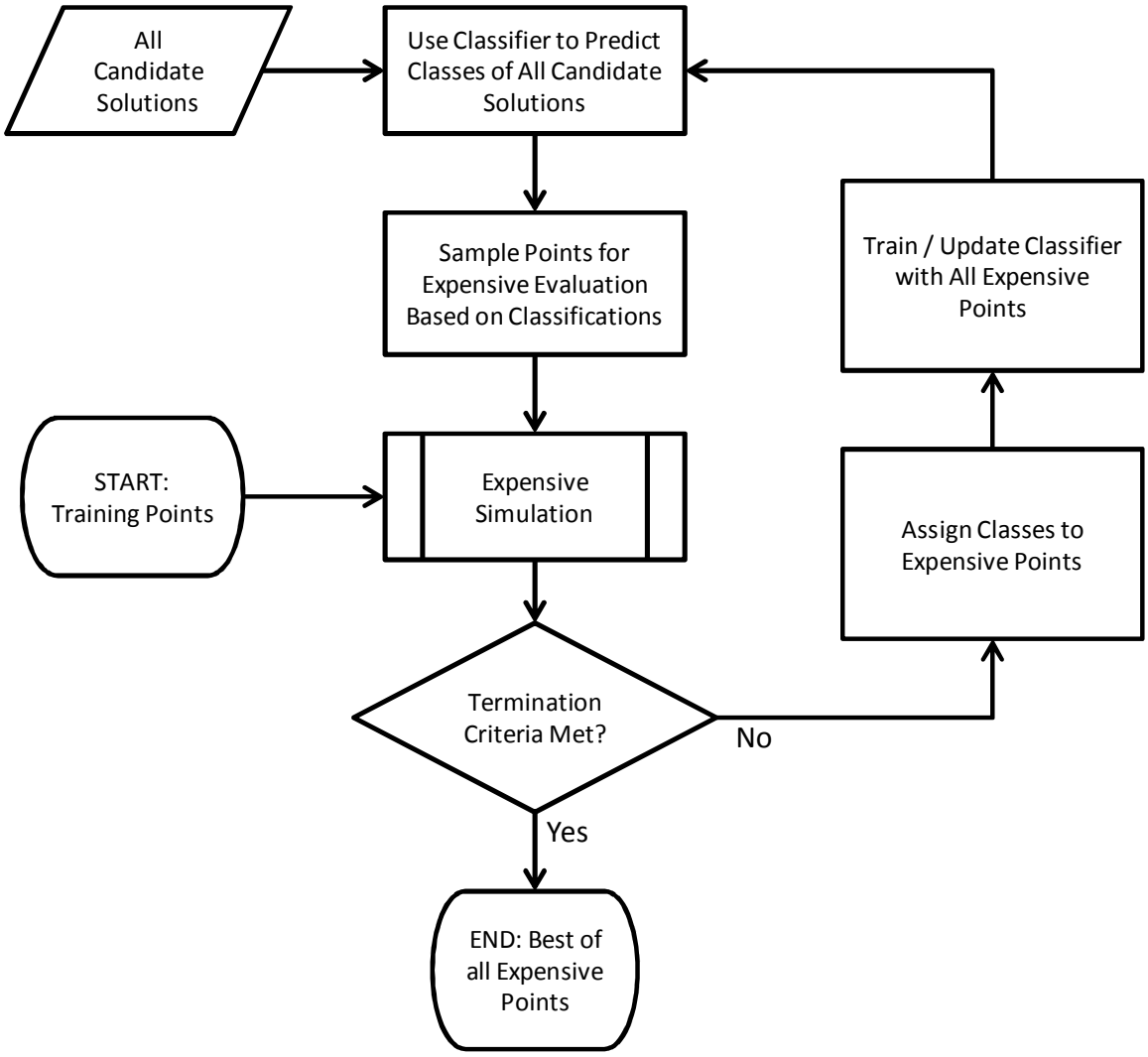


Figure 4.8: Classifier-guided sampling method

The class threshold, T_C, can remain fixed throughout the solution process, but performance is improved by allowing it to change as the classifier learns more about the performance space. In this research, T_C is allowed to change to help the classifier better differentiate between 'high' and 'low' quality solutions. When candidate points are evaluated by the classifier, no points are classified as 'high' if there are no points in the training set with this class label. Therefore, T_C is initially set so that 5% of the training points are assigned a class label of 'high' to give the classifier a sample of the

characteristics of higher quality solutions. As more training points become available from expensive function evaluations and the current best known solution improves, the threshold is made more stringent to reduce the number of candidate points that are assigned a class label of 'high'. Conversely, if zero or very few candidate points are classified as 'high' at an intermediate iteration, T_C is made less stringent to allow more points to be allocated to the 'high' class. This strategy of changing T_C is intended to avoid fixation on suboptimal solutions or local minima. The effect is similar to that of an annealing or cooling schedule in simulated annealing algorithms.

The classifier is trained using training point pairs that include both the training point design variable values and the corresponding class labels. After training, the classifier is used to predict the classes of all candidate solutions. For each unexplored point in the set of candidate solutions, the classifier returns the class label as an output. These so-called 'cheap points' provide categorical predictions of the quality of all candidate solutions that have not been evaluated with the expensive simulation.

Once the set of cheap points is generated using the classifier, their class labels and posterior probabilities, $P(c_k|\mathbf{x})$, are used to determine which unexplored points are to be sampled for the next batch of expensive simulations. In general, priority should be given to the points that are classified as 'high'. However, it is also important, especially early in the solution process, to sample points throughout the entire design space to improve classifier accuracy. The classifier is trained with a relatively low number of points, and it may not predict high quality solutions reliably without more knowledge of the design space. Therefore, three types of points are designated for sampling to achieve the necessary balance between depth and breadth of the search: high-certainty high-class points, high-certainty low-class points, and uncertain points. The high-certainty points are those with posterior probabilities greater than 0.6 for that particular class. For

example, a point for which $P(c_{high}|\mathbf{x}) = 0.75$ and $P(c_{low}|\mathbf{x}) = 0.25$ is considered a high-certainty high-class point. Likewise, a point with $P(c_{high}|\mathbf{x}) = 0.10$ and $P(c_{low}|\mathbf{x}) = 0.90$ is considered a high-certainty low-class point. However, a point for which $P(c_{high}|\mathbf{x}) = 0.55$ and $P(c_{low}|\mathbf{x}) = 0.45$ would be considered an uncertain point. Uncertain points can be from either class, as long as the maximum $P(c_k|\mathbf{x})$ of both classes is less than or equal to 0.6.

These three types of points are strategically sampled to infuse both depth and breadth into the sampling step of the CGS method. Let N_s be the number of points to be sampled during each iteration. In most cases, N_s is composed entirely of high-certainty high-class points and uncertain points. The percentage of N_s that is composed of high-certainty high-class points changes with each iteration, and the initial percentage is a user-defined parameter. This percentage is increased linearly to a maximum of 90% with subsequent iterations, as shown in Figure 4.9. If several iterations occur with no improvement to the current best known solution, the percentage of N that is composed of high-certainty high-class points is adjusted to 50% or below and continues to increase linearly to 90% with each subsequent iteration. High-certainty high-class points are sampled randomly up to the prescribed proportion of N_s, and the remaining N_s points are selected randomly from the pool of cheap points that meet the criteria for uncertain points. Sampling of the third type of point, the high-certainty low-class points, is rare. These points are only sampled in the unusual event that the total number of high-certainty high-class points and uncertain points in the entire pool of cheap points is less than N_s. In this case, the high-certainty low-class points are sampled randomly until there are N_s points selected for expensive simulation in the next step of the CGS algorithm.

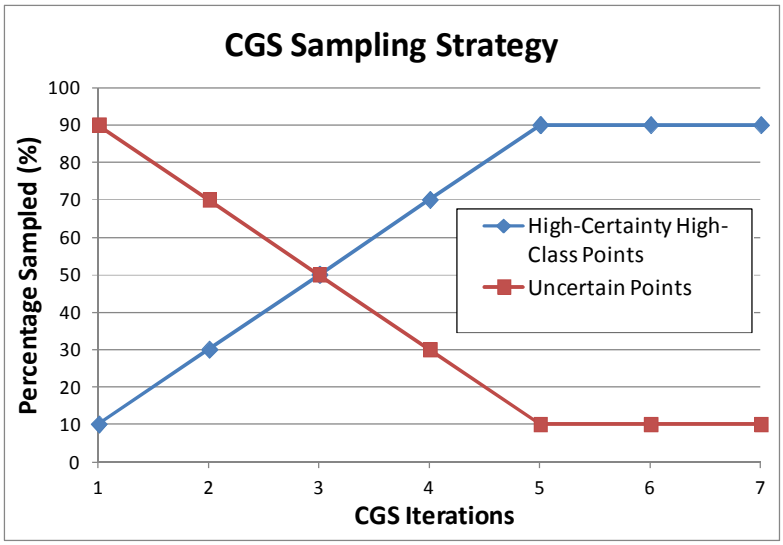


Figure 4.9: Uncertain point versus high-certainty high-class point sampling strategy

Once the expensive simulations are performed, termination criteria are evaluated. Some options for termination criteria include upper limits on the number of expensive simulation evaluations or algorithm iterations, or achievement of a desired objective function value. The CGS algorithm repeats until the termination criteria are met, at which point the best known solution is provided as the output.

The CGS method shares many of the same characteristics as the discrete variable mode pursuing sampling (D-MPS) method that is reviewed in Chapter 2. Both methods begin with a set of candidate solutions that are evaluated using the expensive base function evaluation. Both methods use these evaluations to train and use an inexpensive approximation in an attempt to reduce the total number of expensive base function evaluations. Lastly, both methods use the approximation to guide the search towards regions of the design space where high quality solutions are believed to exist.

The key difference between the D-MPS method and the CGS method is that a classifier is used in the proposed method to provide categorical estimates of the fitness of candidate solutions, while a metamodel is used in the D-MPS method to interpolate the training points and provide quantitative estimates of the fitness of candidate solutions. A metamodel would not be appropriate for discrete variable problems, such as the all-electric ship design and configuration problem, due to the discontinuous, non-differentiable nature of the response. However, a classifier is appropriate because classifiers are able to accept discrete variable inputs and provide categorical outputs that are discontinuous by nature.

The CGS method is a direct search optimization algorithm, similar to simulated annealing, genetic algorithms, and tabu search. That is, it requires no explicit knowledge of an analytical objective function. The key difference between the CGS method and the three methods reviewed in Chapter 2 lies in its use of a classifier to reduce the total number of evaluations of the expensive base simulation. The CGS method uses a classifier as a substitute, or surrogate of the base model. Although the classifier is unable to provide a quantitative estimate of the objective function output, it can give a qualitative estimate of the fitness of a candidate solution. Therefore, the hypothesized advantage of the CGS method is that it is able to reduce expensive base function evaluations by limiting expensive simulations to points that are suggested by the classifier to result in favorable objective function values.

The CGS method is a tool for rapid design space exploration for finding acceptable solutions quickly. It is best described as a direct sampling metamodel-based design method for discrete variable / discontinuous response problems, in which a classifier is used in place of a traditional continuous variable metamodel. In the next

section, the performance of the CGS method is compared to genetic algorithms when applied to three discrete variable optimization problems.

4.3 CLASSIFIER-GUIDED SAMPLING PERFORMANCE EVALUATION

In this section, the CGS method is applied to three discrete variable discontinuous response optimization problems. Genetic algorithms (GA's) and random search are also applied to the three test problems and performance comparisons are made. The three test problems are explained in Section 4.3.1. In Section 4.3.2, the encodings, selection method, crossover operators, mutation operators, and constraint handling approach for the GA implementations are given. In Section 4.3.3, the method for selecting the user-defined parameters for the CGS method and the GA is discussed. Lastly, the results of the tests and a discussion of these results are presented in Section 4.3.4.

4.3.1 Test Problems

Three optimization problems are solved using the CGS method, GAs, and random search. The first two problems, a knapsack problem and a traveling salesman problem, are common combinatorial optimization problems. The third problem, the welded beam design problem, is a mechanical engineering design problem. These problems are selected to provide a broad range of problem types to test the classifier-guided sampling method.

Test Problem 1: 20-Item Knapsack Problem

The objective of the knapsack problem is to select items from a set of available items that will maximize the combined value, V, of all selected items without exceeding a total weight limit, W. In the knapsack problem used in this study, there are 20 different items, and there is only one of each type of item available for selection. Denoting a

vector of binary variables $\mathbf{x} = (x_1, x_2,\ldots,x_{20})$ to represent the selection of items, the problem is formulated as follows:

$$\text{Maximize} \quad V(\mathbf{x}) = \sum_{i=1}^{n} v_i x_i \tag{4.12}$$

$$\text{Subject to} \quad \sum_{i=1}^{n} w_i x_i \leq W,\ x_i \in \{0,1\} \tag{4.13}$$

where v_i and w_i are the value and weight of item i, respectively. W is chosen to be 50% of the total weight of all available items. The weights and values of the 20 available items are given in Table 4.2:

Table 4.2: 20-Item knapsack problem parameters

Item	*Weight* (w_i)	*Value* (v_i)
1	94	3
2	70	41
3	90	22
4	97	30
5	54	45
6	31	99
7	82	75
8	97	76
9	1	79
10	58	77
11	96	41
12	96	98
13	87	31
14	53	28
15	62	58
16	89	32
17	68	99
18	58	48
19	81	20
20	83	3

Test Problem 2: 11-City Traveling Salesperson Problem

The traveling salesperson problem (TSP) is a frequently studied combinatorial optimization problem. Given a set of cities and their Cartesian coordinates, the objective is to find the shortest possible tour that visits each city exactly once and returns to the city of origin. If there are n cities to visit, there are $n!$ possible solutions to this problem. The problem size is reduced to $(n\text{-}1)!/2$ by specifying a city of origin at which the tour will always begin and end and by assuming symmetry, *i.e.* the distance between any two cities is the same regardless of direction traveled. A tour is represented by a vector of integer variables $\mathbf{x} = (x_1, x_2,\ldots,x_{10})$ where x_i is an integer from one to ten and each integer value can appear only once in each solution. By specifying the variables in this way, the objective function can be formulated as:

$$\text{Minimize} \quad D(\mathbf{x}) = d_o(x_1) + \sum_{i=1}^{n-2} d(x_i, x_{i+1}) + d_o(x_{10}) \tag{4.14}$$

where d is the Euclidean distance between cities x_i and x_{i+1} and d_o is the distance from the city of origin to the first or last city in the tour. The Cartesian coordinates of the origin and the 10 tour cities are given in Table 4.3:

Table 4.3: 11-City traveling salesperson problem city coordinates

City	*Abscissa*	*Ordinate*
Origin	14.8	42.7
1	98.6	29.8
2	5.0	4.5
3	39.9	12.0
4	28.0	22.6
5	57.0	79.2
6	26.2	90.7
7	67.9	55.1
8	76.0	47.1
9	86.4	67.5
10	47.3	98.5

Test Problem 3: Welded Beam Design Problem

The welded beam design problem, adapted from [102] and [103], is an engineering optimization problem that combines categorical and quantitative discrete variables. A rectangular bar is welded at one end and serves as a cantilever beam to carry a point load at the opposite end. The objective is to select the weld type, material, and geometric parameters that minimize the cost of fabrication. The two weld types and geometric parameters are shown in Figure 4.10.

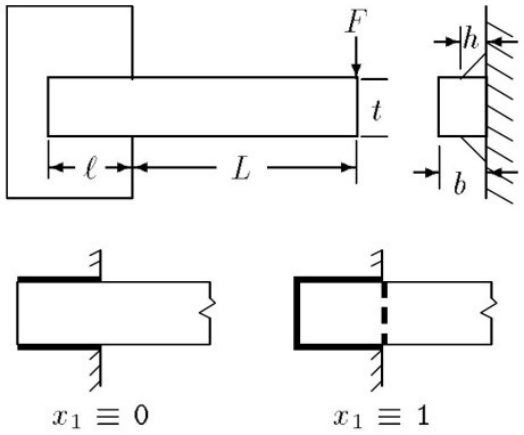


Figure 4.10: Welded beam problem [103]

There are a total six variables that compose a solution to this problem. The weld configuration is binary and describes whether two ($x_1 = 0$) or four ($x_1 = 1$) of the contact edges between the beam and base are to be welded. The weld and beam material is represented by one of four integers: $x_2 = 1$ (steel), $x_2 = 2$ (cast iron), $x_2 = 3$ (brass), and $x_2 = 4$ (aluminum). The geometric parameters are the thickness of the weld ($x_3 = h$), the width of the beam ($x_4 = t$), the thickness of the beam ($x_5 = b$), and the length of the welded portion of the beam ($x_6 = l$). The variables that describe the geometric parameters are restricted to a finite set of discrete values (Table 4.3):

Table 4.3: Welded beam geometric parameter ranges

Variable	*Symbol*	*Min (in.)*	*Max (in.)*	*Step Size (in.)*
x_3	h	0.0625	0.5000	0.0625
x_4	t	7.500	10.000	0.125
x_5	b	0.0625	1.0000	0.0625
x_6	l	0.125	3.000	0.125

If the six design variables described above are represented by the vector $\mathbf{x} = (x_1, x_2, x_3, x_4, x_5, x_6)$, the objective function and constraints are given by

$$\text{Minimize} \quad f(\mathbf{x}) = (1+c_1)x_3^2(x_6 + x_1x_4) + c_2x_4x_5(L + x_6) \tag{4.15}$$

$$\text{Subject to} \quad \begin{cases} g_1(\mathbf{x}) = S - \sigma(\mathbf{x}) \geq 0 \\ g_2(\mathbf{x}) = P_c(\mathbf{x}) - F \geq 0 \\ g_3(\mathbf{x}) = \delta_{\max} - \delta(\mathbf{x}) \geq 0 \\ g_4(\mathbf{x}) = 0.577S - \tau(\mathbf{x}) \geq 0 \end{cases} \tag{4.16}$$

where c_1 and c_2 are material costs and g_1, g_2, g_3, and g_4 are constraints on the bending stress $\sigma(\mathbf{x})$, buckling load $P_c(\mathbf{x})$, beam deflection $\delta(\mathbf{x})$, and weld shear stress $\tau(\mathbf{x})$, respectively. The force of the load F is 6,000 lb., the extended length L of the beam is 14 in., and the maximum allowable deflection $\delta_{\max}$ is 0.25in. The material costs, properties and constraint equations are provided in Appendix A.

Test Problem Global Optima

For benchmarking purposes, the global optima for each test problem are provided in Table 4.4. The 20-item knapsack problem has two optima with identical objective function values, while the TSP and welded beam problem have single global optima.

Table 4.4: Test problem global optima

Knapsack Problem		
$\mathbf{x}^*$	W^*	V^*
[0 0 0 0 1 1 1 1 1 1 1 1 0 0 1 0 1 1 0 0]	703	795
[0 1 0 0 1 1 1 1 1 1 0 1 0 0 1 0 1 1 0 0]	677	795
Traveling Salesperson Problem		
$\mathbf{x}^*$	D^*	
[2 4 3 1 8 7 9 5 10 6]	333.17	
Welded Beam Design Problem		
$\mathbf{x}^*$	g_i^*	f^*
$x_1 = 1$ $x_2 = 1$ (steel) $x_3 = 0.1875$ in. $x_4 = 8.25$ in. $x_5 = 0.25$ in. $x_6 = 1.75$ in.	$g_1 = 380.165$ psi $g_2 = 402.047$ lb. $g_3 = 0.2344$ in. $g_4 = 244.567$ psi	\$1.9509

4.3.2 Genetic Algorithm Implementation

In this section, the encodings, selection method, crossover operations, mutation operations, and constraint handling methods are discussed for the genetic algorithm (GA) solution to the three test problems described in Section 4.3.1. Since their initial conception in the 1970's, numerous modifications and problem specific algorithms have been developed in the GA research community. An exhaustive study of the ways in which GA could be used to solve the three test problems in Section 4.3.1 is outside the scope of this work at this time. Therefore, the traditional GA implementation describe by Goldberg [64] is used to the maximum extent possible in this research. Exceptions are made for the traveling salesperson problem (TSP) encoding, crossover, and mutation methods due to the high occurrence of illegal tours (*e.g.*, the same city visited more than once in a tour) that result when the traditional crossover and mutation methods are used.

Of the three discrete variable optimization methods reviewed in Chapter 2, simulated annealing (SA) and tabu search (TS) are drastically different than the CGS

method whereas GAs have some similar characteristics. SA and TS evaluate and compare only one solution at a time, whereas the CGS method and GA both evaluate and compare a pool of solutions at each iteration. Another distinction can be made with regard to the manner in which the methods explore new solutions. With SA and TS, each new candidate solution is always a single "move" away from the current solution. GAs explore new solutions by combining previous ones so that "children" of "parent" solutions are meaningfully created and properties of good solutions are carried into subsequent generations. The CGS method is similar in the sense that, with adequate training points, the classifier recognizes good solution properties and therefore classifies candidate solutions with similar properties as good solutions. Lastly, GAs and the CGS method both begin with an initial set of candidate solutions, which serves as the classifier training points for CGS or the initial population for GAs. For the reasons described above, GAs are selected for comparison to the proposed method.

Encodings

In a traditional GA implementation for multivariable design optimization problems, variables are mapped into binary form and concatenated into a longer string to form a candidate solution. For the knapsack problem, the 20 variables are already binary, *i.e.* an item either is or is not included in the solution. Therefore, no additional encoding is necessary for the knapsack problem. For the TSP, most crossover operations result in illegal tours when traditional binary encodings are used [104]. Therefore, no additional encoding is used for the TSP in this research, and a candidate solution remains in the form described in Section 4.3.1, also known as *path representation* [104].

The encodings for the welded beam problem are less straightforward than those for the knapsack problem and TSP. Most of the variables are not binary, but unlike the

TSP, a binary encoding of the variables in the welded beam problem lends itself nicely to traditional crossover and mutation operators. Therefore, the welded beam problem design variables are coded into binary form using multiparameter, mapped, fixed-point coding, as described in [64]. A total of 20 bits are needed to achieve the necessary mapping precision between decimal and binary form for the welded beam problem.

Selection

There are several ways to perform the selection step of a GA including tournament selection, ranking selection, and Genitor (or "steady state") [105]. In this research, the traditional "roulette wheel" selection method [64] is used. With roulette wheel selection, each member of the current population has a roulette wheel section, or slot, whose size is proportional to its fitness. An individual with a larger slot has a higher chance of being selected for reproduction into the next generation. The size of each member's slot is determined by dividing that member's fitness by the sum of the fitness of all members in the current population. Lastly, elitism is used in the GA implementation presented here. In each generation, the N_e best solutions (elites) are always passed along to the next generation without being mated or mutated, regardless of whether they are selected for reproduction. N_e is either equal to two (2) or 5% of the population size, whichever is greater.

Crossover Operators

For the knapsack problem and the welded beam problem, "simple crossover" is used during the mating phase of the GA. In simple crossover, members of the new population resulting from the selection phase are paired at random to serve as parents of new child strings. An integer k between 1 and the string length minus one is selected uniformly at random. The two children are created by swapping all bits in the string that

come after position k in both parent strings. For example, Figure 4.10 shows a simple crossover operation for two 8-bit strings with $k = 4$.

$$\begin{array}{l} parent_1 = [1 \;\; 1 \;\; 0 \;\; 1 \;|\; 0 \;\; 0 \;\; 1 \;\; 1] \\ parent_2 = [0 \;\; 1 \;\; 1 \;\; 0 \;|\; 1 \;\; 0 \;\; 1 \;\; 0] \end{array} \longrightarrow \begin{array}{l} child_1 = [1 \;\; 1 \;\; 0 \;\; 1 \;|\; 1 \;\; 0 \;\; 1 \;\; 0] \\ child_2 = [0 \;\; 1 \;\; 1 \;\; 0 \;|\; 0 \;\; 0 \;\; 1 \;\; 1] \end{array}$$

Figure 4.10: Simple crossover for 8-bit binary strings

For the TSP, simple crossover is not the best option because this operation often generates illegal tours in which the some cities may be visited more than once and other cities are not visited at all. For example, when simple crossover is performed on the two parents shown in Figure 4.11, the two child tours that result are both non-permissible tours.

$$\begin{array}{l} parent_1 = [1 \;\; 2 \;\; 3 \;\; 4 \;|\; 5 \;\; 6 \;\; 7 \;\; 8] \\ parent_2 = [7 \;\; 3 \;\; 6 \;\; 1 \;|\; 8 \;\; 5 \;\; 2 \;\; 4] \end{array} \longrightarrow \begin{array}{l} child_1 = [1 \;\; 2 \;\; 3 \;\; 4 \;|\; 8 \;\; 5 \;\; 2 \;\; 4] \\ child_2 = [7 \;\; 3 \;\; 6 \;\; 1 \;|\; 5 \;\; 6 \;\; 7 \;\; 8] \end{array}$$

Figure 4.11: Simple crossover resulting in illegal TSP tours

While it is possible to perform repair operations to remedy this situation (*i.e.* replacing a duplicate city with a nonexistent one), it is much simpler to use an alternate crossover operation that precludes the possibility of an illegal child tour. Therefore, in this study, order crossover [106,104] is selected as the crossover operation to be used. With order crossover, offspring are created by choosing two cut points, k_1 and k_2, uniformly at random. The two cut points must not be equal, and they must be integers between one and the length of the string minus one. For both parents, the subtours that fall between the two cut points are copied directly to the children. The remaining cities in one child are generated by copying them from the other parent in the order that they

appear, starting at the second cut point. When the end of the tour is reached, cities are then copied from the beginning of the tour. Any cities that already exist in the child tour are skipped. An example of order crossover for two 8-city tours with $k_1 = 2$ and $k_2 = 5$ is shown in Figure 4.12. In the example, the subtours in the boxes are copied directly from parents to their respective children. The remaining cities are copied in the order that they appear in the opposite parent, starting after the copied subtour, and omitting any cities that already exist in the tour.

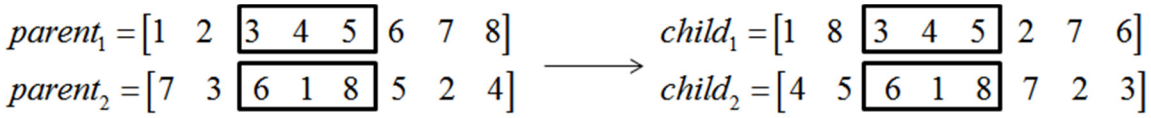


Figure 4.12: Order crossover example

Order crossover is an ideal operation for encoding the path representation of the TSP for several reasons. First, if executed correctly, it is impossible to produce illegal tours as offspring. Second, it preserves the two attributes of a TSP solution that affect fitness: the order of the cities and the absolute positions of the cities. Order crossover effectively passes both of these attributes from parents to children, thus supporting the fundamental GA notion of survival of the fittest.

Mutation Operators

For the knapsack and welded beam problem, the traditional mutation operator is used, which involves choosing one bit in the binary solution string at random and reversing its value, *i.e.* a 1 is changed to a 0 or vice versa. For the TSP, this operation suffers from the same challenge as the traditional simple crossover operation, potentially resulting in solution strings that represent illegal tours. Therefore, exchange mutation [104] is used in this study as the TSP mutation operator. In exchange mutation, two cities

in a tour are selected at random and their positions are swapped. This maneuver always results in a legal tour that is very similar to the original, pre-mutated tour.

Constraints

The knapsack and welded beam problem are examples of constrained optimization problems in which the objective function must be optimized without violating one or more constraint equations. Throughout the GA solution process, crossover and mutation operations often produce infeasible offspring that violate these constraints. In this research, constraint violations are handled with the use of a penalty function. A penalty function transforms the constrained problem into an unconstrained one by penalizing solutions that are infeasible [64]. The most common type of penalty functions are those that penalize the objective function according to a user-defined level of constraint violation. The penalty function used in this research is similar to the approach suggested by Morales and Quezeda [107], which is based on the number of constraints violated by an infeasible solution (Equation (4.16)):

$$f_{new} = f_{old} + K \bullet N_{cv} \tag{4.16}$$

where f_{new} is the penalized fitness value, f_{old} is the infeasible fitness value, K is a user-defined constant, and N_{cv} is the number of constraints violated.

4.3.3 User-Defined Parameter Selection

In both the CGS method and the GA solution to the three test problems, the performance of the method depends in part on a handful of user-defined tuning parameters. Special care must be taken when selecting these parameters to ensure that a fair comparison is conducted and presented. Therefore, a preliminary set of parameter tests are conducted for each test problem in order to identify best performing parameter

combinations. The parameter combinations that yield the strongest performance on the preliminary tests are used for the main comparison and validation study.

User-Defined Parameters for Classifier-Guided Sampling

For the CGS solution to the test problems, there are three user-defined tuning parameters that are considered in the parameters selection study. The first parameter is the number of training points in the initial training set, N_{tr}. The second parameter is the number of new points to sample for expensive evaluation at each iteration, N_s. The third parameter is the initial value of the percentage of high-certainty high-class points to sample from the pool of cheap points, P_{hs}. Each parameter is evaluated at three different levels for each test problem, resulting in 27 possible parameter combinations. Each of the 27 parameter combinations are tested five times for each test problem, and the parameter combinations that enable the CGS algorithm to identify a predetermined objective function value with the fewest function evaluations, on average, are selected for inclusion in the main comparison study. The three parameters levels for each test problem are shown in Table 4.5. The parameter values in boldface are those for which the CGS algorithm identifies a predetermined objective function value with the fewest function evaluations on average.

Table 4.5: Classifier-guided sampling user-defined parameters

Test Problem	N_{tr}	N_s	P_{hs}
Knapsack	[50, **100**, 200]	[**50**, 100, 200]	[0.1, 0.5, **0.9**]
Traveling Salesperson	[**100**, 200, 400]	[**100**, 200, 400]	[0.1, **0.5**, 0.9]
Welded Beam	[**50**, 100, 200]	[50, **100**, 200]	[0.1, **0.5**, 0.9]

In addition to the parameters described above, the Bayesian network structure must be selected. For the knapsack and the welded beam problems, the Naïve Bayes (NB) classifier is used in the CGS algorithm. This Bayesian network structure is chosen

because relatively few training points are needed to populate the class conditional probability distributions with adequate density, as discussed in Section 4.2. Furthermore, although the assumption of independence is often incorrect, the Naïve Bayes classifier frequently performs well in practice because its classification decision may be correct even if its class conditional probability estimates are inaccurate [108]. Furthermore, Zhang [109] show that Naïve Bayes is optimal even when dependencies exist if those dependencies cancel each other out.

Due to the strong conditional dependence among variables in the TSP, the NB classifier does not provide adequate classification accuracy, and the CGS method fails to converge towards the known global optimum in some trials. This result is directly related to the strong assumption of independence built into the Naïve Bayes classifier. In light of the above discussion, an Augmented Naïve Bayes (ANB) network structure [109] is used for the TSP in this research. In the ANB classifier, each variable node is conditionally dependent on its adjacent variable node (Figure 4.13). While this BN structure does not capture the highly dependent nature of the variables in the TSP, it improves performance by enabling the classifier to recognize dependencies between adjacent cities in the TSP tour.

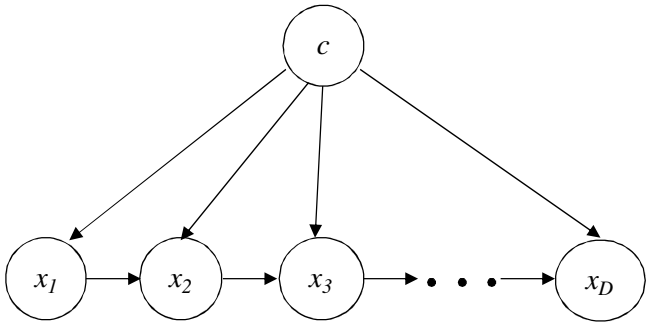


Figure 4.13: Bayesian network structure for the TSP

User-Defined Parameters for the Genetic Algorithm

For the GA solution to the test problems, there are also three user-defined tuning parameters that are considered in the parameter selection study. The first parameter is the population size, N. The second parameter is the percentage of non-elite population members that are selected for reproduction to mate with crossover operations, P_c. The third parameter is the percentage of encoded bits to mutate in each generation, P_{hs}. As is done for the CGS parameter selection study, each parameter is evaluated at three different levels for each test problem, resulting in 27 possible parameter combinations. Each of the 27 parameter combinations are tested ten times for each test problem, and the parameter combinations that enable the GA to identify a predetermined objective function value with the fewest function evaluations, on average, are selected for inclusion in the main comparison study.

Table 4.6: Genetic algorithm user-defined parameters

Test Problem	N	P_c	P_m
Knapsack	[25, **50**, 100]	[0.6, 0.8, **1**]	[0.005, 0.010, **0.015**]
Traveling Salesperson	[25, **50**, 100]	[**0.6**, 0.8, 1]	[0.005, **0.010**, 0.015]
Welded Beam	[25, 50, **100**]	[0.6, **0.8**, 1]	[0.005, **0.010**, 0.015]

The three parameters levels for each test problem that were considered in the parameters selection study are shown in Table 4.6. The parameter values in boldface are those that resulted in the best average performance.

4.3.4 Results and Discussion

Results

The performance of the CGS method and the GA described in the previous section is evaluated by executing a set of rate of convergence tests. In a rate of convergence test, the current best solution versus the number of objective function

evaluations is recorded. This test provides a visual measure of how quickly each method converges towards known global optima.

A purely random search is also performed on the three test problems to provide an additional comparison. For the random search method, each new trial begins with the set of all possible solutions. Samples are randomly pulled from the exhaustive solution space without being replaced to avoid repeat evaluations of identical solutions. As the iterations proceed, the solution with the best objective function value is retained until a solution with a better objective function value is randomly sampled.

Random number generation is a key element of the solution process in all three of the compared methods, and performance varies with each repeat solution of the same test problem. Therefore, the convergence rate tests are repeated 50 times for each method on all test problems, and the primary performance comparison is based on the mean performance. In addition to the mean performance, the 10^{th} and 90^{th} percentiles of all 50 tests are calculated at select points to provide an indication of the extent to which performance varies about the mean.

The results for the rate of convergence tests for the knapsack, traveling salesperson, and welded beam problems are shown in Figures 4.14 – 4.16. The error bars at select function evaluation values represent the $10^{th}/90^{th}$ percentiles. The percentiles are slightly offset from each other to prevent overlapping and improve visual clarity.

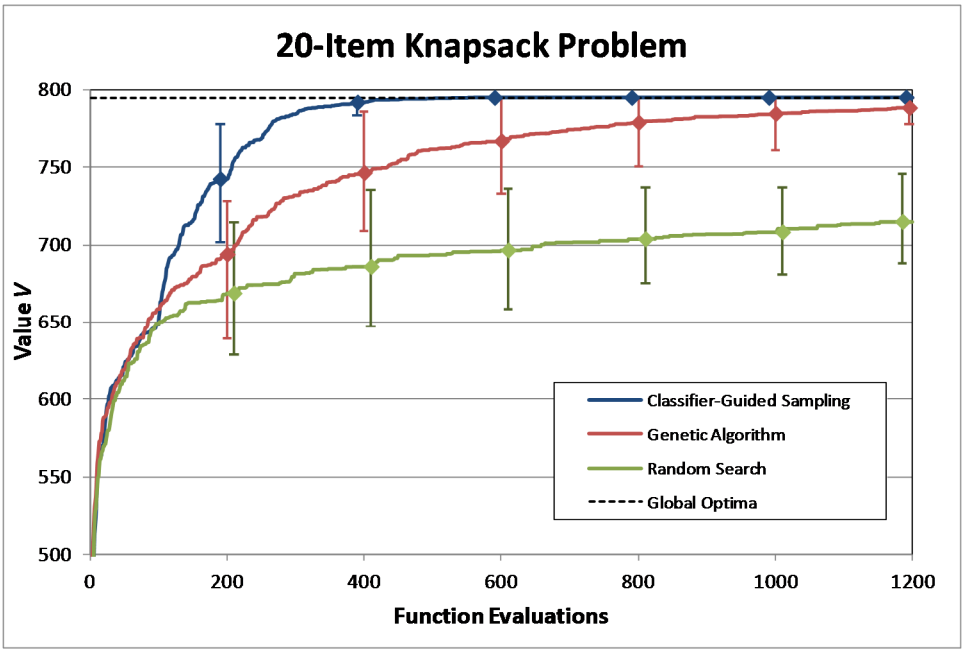


Figure 4.14: Knapsack problem convergence test results with $10^{th}/90^{th}$ percentile bars

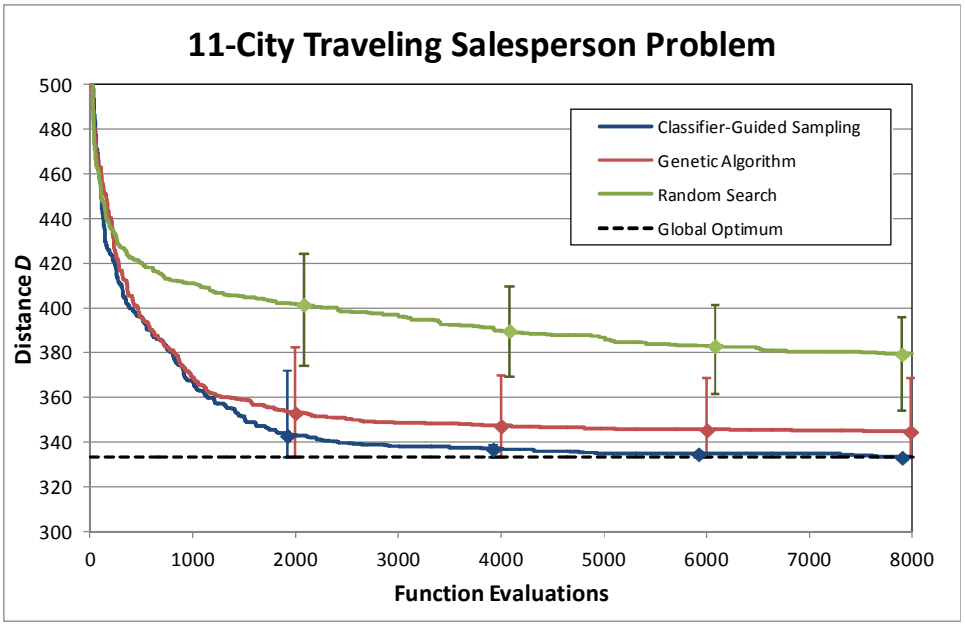


Figure 4.15: 11-city TSP convergence test results with $10^{th}/90^{th}$ percentile bars

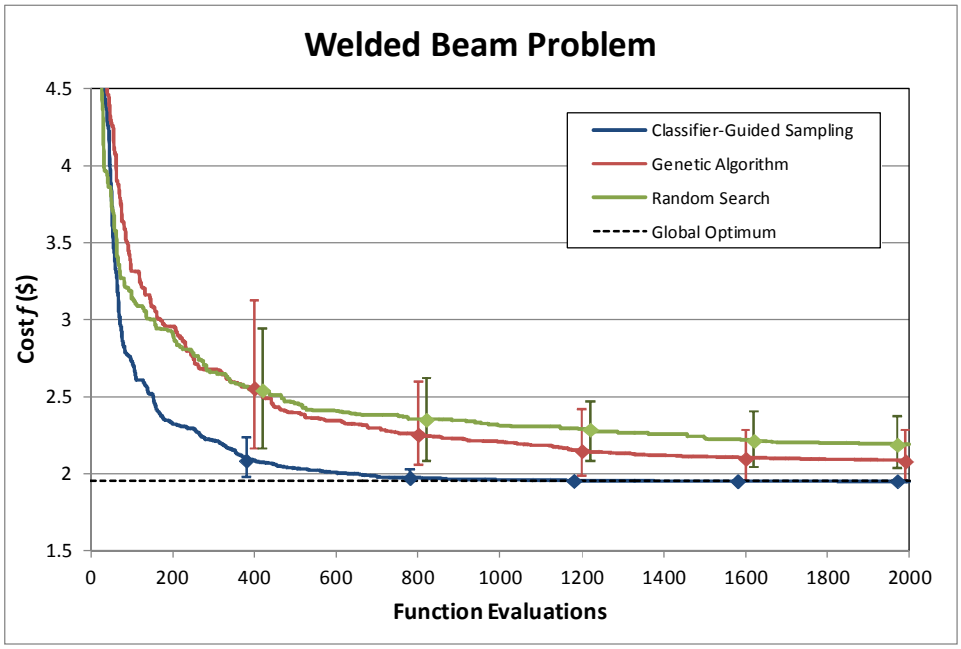


Figure 4.16: Welded beam problem convergence test results with 10^{th}/90^{th} percentile bars

Initially, the CGS method performs no better than random search. This behavior is expected because the first batch of points selected for function evaluation is selected randomly to serve as the initial training set. Once the CGS becomes active after evaluating the initial training set (100 function evaluations for the knapsack and TSP, 50 function evaluations for the welded beam), the average best known solution approaches the known optimum rapidly compared to random search. This behavior can be seen on all three of the test problems.

For the 20-item knapsack problem (Figure 4.14), the CGS method converges on one of the two global optima with just over 400 function evaluations, while it takes the GA over 1,200 function evaluations, on average. In addition to faster average rate of convergence, the CGS method is less prone to fixation on suboptimal solutions. One of the two global optima is found with the CGS method with fewer than 1,200 function

evaluations in all 50 of the trials. However, in three of the 50 trials the GA fails to find the optimal solution in fewer than 2,500 function evaluations.

For the 11-city TSP (Figure 4.15), the performances of the CGS method and the GA are similar up to about the first 1000 function evaluations. However, beyond this point the GA begins to flatten out while the CGS method continuous to converge towards the optimal solution. The CGS method finds the global optimum with fewer than 3,000 function evaluations in the majority of the 50 trials, and in all 50 trials it does so in fewer than 8,000 function evaluations. However, in many of the trials the GA fails to converge on the global optimum with fewer than 10,000 function evaluations.

For the welded beam problem (4.16), the CGS method drastically outperforms the GA, and the GA only marginally outperforms random search. In most cases the CGS method identifies the global optimum in fewer than 800 objective function evaluations. However, in three of the 50 trials the CGS method fails to find the optimum with fewer than 2,000 function evaluations. The GA struggles with the welded beam problem; the average best solution it identifies in 2,000 function evaluations is equal to that which the CGS method is able to find with about 450 function evaluations.

The 10^{th}/90^{th} percentile bars exhibit similar behavior for all three test problems. Early in the solution process, the CGS solutions in the 10^{th} percentile are inferior to the GA solutions in the 90^{th} percentile. This trend indicates that if a limited number of function evaluations are allowed, the CGS method would outperform the GA in most, but not all cases. However, as the number of function evaluations increases, the CGS solutions that are in the 10^{th} percentile are superior or equal to the GA solutions that are in the 90^{th} percentile. Therefore, given an adequate number of function evaluations, the CGS method will perform better than the GA in most cases, and equal to the GA in a few select cases.

Discussion

The results show that the performance of the CGS method is substantially better than the GA and random search in all test cases. For all three test problems, the CGS method converges to the known global optimum with significantly fewer objective function evaluations. These results support the hypothesis that a discrete variable classifier can be used in place of a continuous variable metamodel to solve computationally expensive, discrete variable / discontinuous response design problems.

The CGS method is particularly effective at solving the 20-item knapsack problem (4.14) despite the large number of variables and possible solutions when compared to the welded beam problem. This performance can be partially attributed to the fact that the variables affect the objective function independently of each other. Because of this conditional independence, the Naïve Bayes classifier is well suited to predict the quality of unexplored test points. For example, referring to Table 4.2, items 6, 9, 10, 12, and 17 are all high "value" items. Setting the variables that correspond to these items equal to $x_i = 1$ always has a strong, favorable effect on the objective function regardless of the values of the other variables. Therefore, the Naïve Bayes classifier is very well suited to predict solution quality for this problem because it assumes the variables are conditionally independent.

The CGS method has the most difficulty solving the 11-city TSP. There are two reasons for this result. First, this problem has about 1.8 million possible solutions, compared to only about 1 million for the knapsack problem and about 516,000 for the welded beam problem. Second, the variables in this problem have strong conditional dependence on each other because the order, or pattern, of the cities in the tour has a greater effect on the objective function than the absolute positions of the cities in the tour. For instance, in select trials the CGS method becomes fixated on the solution $\mathbf{x}$ = [3 1 8 7

9 5 10 6 2 4] for many iterations before finding an improving tour. This solution is very similar to the global optimum in which cities 2 and 4 are the first two cities visited instead of the last two: **x*** = [2 4 3 1 8 7 9 5 10 6]. Therefore, moving from the near optimal solution to the optimal solution in this case requires that *all* of the values of **x** change. The classifier had been trained with a large number of 'high' class points with short distance subtours, such as $x_1 = 3$, $x_2 = 1$, $x_3 = 8$, and $x_4 = 7$, and so on, and it therefore mistakenly classifies other solutions with these particular variable values as 'high'. In such cases, the classifier may view the true optimum as categorically different from the local minimum and misclassify the true optimum as 'low'. When a multinomial distribution is used to estimate the class conditional probabilities, the classifier is based primarily on the frequency that a particular variable takes on a certain discrete value. In the case of the TSP, these frequencies correspond to the absolute positions of the cities in the tour. However, the classifier does not necessarily recognize solutions that contain strong patterns, or subtours, as being high quality. Sampling points from the 'uncertain' category, as described in Section 4.2, are found to significantly improve classifier accuracy and reduce the likelihood of this situation occurring. Using the augmented Naïve Bayes classifier (ANB), as opposed to the standard Naïve Bayes classifier, also significantly improves performance for this problem.

Since the ANB classifier improves performance over the Naïve Bayes classifier for the TSP, it is logical to think that a fully connected Bayesian network classifier would improve performance even further. However, the multinomial probability distribution that results from a fully connected Bayesian network is a large, multidimensional space that could not possibly be adequately populated with training points. The number of bins in the discrete variable multinomial distribution is exactly equal to the total number of possible tours. Therefore, each bin in the distribution can only have either zero or one

training point counts. Such a classifier would be essentially useless, because all unexplored test points have equal class conditional probabilities across all classes, and classification is based solely on the prior probability (percentage of training points in each class). This situation is unique to the multinomial distribution used for the discrete variable problems that are the focus of this research, and fully connected Bayesian classifiers are frequently used with continuous underlying distributions such as the Gaussian.

The CGS method also performs very well when solving the welded beam problem (Figure 4.16), despite the fact that the variables are not conditionally independent. The dependence of the variables in the welded beam problem is much weaker than in the TSP. First, the variables in the welded beam problem can change independently; changing the value of one variable in the welded beam problem does not require a change in the value of another variable as with the TSP. Second, even though the geometric parameters in this problem are not conditionally independent, their overall effect on the objective function is relatively monotonic: their values should be as small as possible without violating constraints. This fact holds true regardless of weld type and material choice. Lastly, steel is considerably cheaper than the other materials with the exception of cast iron. However, steel has significantly higher yield stress and modulus properties. Accordingly, the classifier requires relatively few training points to learn that any feasible solution that uses steel has a much higher likelihood of having a lower dollar cost (preferred objective function value) than any solution that uses one of the other materials.

Although the CGS method is effective for solving the test problems presented in this study, the number of initial training points and the average number of function evaluations required to find the optimal solution are high. A minimum of several hundred function evaluations is required to achieve convergence to a global optimum in

the problems tested here. If hundreds of function evaluations are required for accurate classification and convergence toward a global optimum, then the method would be useful only for problems of moderate computational expense (i.e., with execution times on the order of minutes, rather than days).

Convergence to a precise known global optimum is not the only measure of optimization algorithm performance. In many practical situations, users are tasked with identifying performance-improving solutions in a limited amount of time. In such cases, a more meaningful measure of optimization algorithm performance is how close the algorithm can get to the global optimum with a fixed number of allowable objective function evaluations. By this measure, the CGS method performs favorably for all three of the test problems presented here. In all three test cases, the CGS method finds solutions that are considerably better than those found by the GA. For the knapsack and welded beam problems, the CGS algorithm finds solutions that were very close to the optima with only 200 function evaluations. For the more difficult traveling salesperson problem, the CGS algorithm finds solutions that are very close to the global optimum in approximately 2,000 function evaluations.

The superior performance of the CGS method compared to the GA can be partially attributed to the lack of repeated objective function evaluations for identical solutions. Repeat evaluations are completely avoided with the CGS method for the particular problems solved here, because all possible candidate solutions can be enumerated in the initial list of unevaluated points, also referred to as "cheap points." When a cheap point is sampled for expensive evaluation with the guidance of the classifier, it is removed from the list of cheap points and appended to the list of expensive points that are used to train the classifier. Therefore, it is impossible for previously sampled points to be sampled and evaluated with the objective function more than once.

In contrast to the CGS method, repeat evaluations are a common occurrence with the traditional implementation of the GA, which is used in this research. For the 50 trials that are performed on the three test problems in this study, there is an *average* of 394.3 repeats during the first 1,200 function evaluations for the knapsack problem, 3,379 repeats during the first 8,000 function evaluations for the TSP, and 802.5 repeats during the first 2,000 function evaluations for the welded beam problem. The cause of many of these repeat evaluations is that, in several of the trials, the GA becomes fixated on a suboptimal solution. When this occurs, the GA is unable to evolve the current population to higher performing solutions because the current generation is largely composed of identical solutions; when two identical solutions are mated during crossover they produce identical offspring. It would have been possible to account for and avoid these repeat function evaluations by keeping a memory of all previously evaluated solutions. However, doing so would have been an unfair comparison to the CGS. Such an approach would have ignored the fact that the GA is fixated on suboptimal solutions, while tens of thousands of repeat individuals pass from one generation to the next before a specific and unlikely mutation operation introduces an improvement to the current best solution.

A Bayesian classifier gives the CGS method the ability to simultaneously pursue high performance solutions (depth search) and regions of the design space that are uncertain (breadth search), while avoiding points that have a high probability of resulting in low performance. By continuously training and updating the classifier before the start of each new iteration, the CGS method is able to use all of the previously evaluated candidate points to sample points for subsequent objective function evaluation. In contrast, GAs search the solution space using a "survival of the fittest" strategy, in which only solutions that are already known to be high performers are selected to be included in subsequent generations. Therefore, the GA has no memory of previously evaluated

solutions other than a small number of "elites" that were passed on from the previous generation.

In this research, training points are sampled randomly and no effort is made to develop a training set that provides the classifier with a variety of solution classes. In practice, however, every attempt should be made to provide the classifier with a training set that contains solutions from all qualitative classes. Even a very large training set will not result in a useful classifier if high quality solutions are absent from the set. Therefore, when generating the initial training set, practitioners should take advantage of any information or intuitions that are available about the problem to be solved. Previous design experience could be used to seed the initial training set with solutions that are known to result in favorable or unfavorable objective function values, thus giving the classifier an early indicator of the characteristics of high quality solutions.

4.4 CLASSIFIER-GUIDED SAMPLING TIME COMPLEXITY

The CGS method identifies high-quality solutions with fewer objective function evaluations than genetic algorithms when applied to the three test problems in the previous section. When the objective function is expensive to evaluate, the time savings of the CGS method are considerable. However, there is additional computational expense associated with the process of training the classifier, classifying the unevaluated test points, and sampling for subsequent expensive evaluation. In most cases, these operations require more computation time than the GA selection, crossover, and mutation steps. In this section the time complexity of the CGS method is studied to gain an understanding of when it may and may not be worthwhile, from a time saving perspective, to use the CGS method. If the objective function is inexpensive to evaluate, it may not be worthwhile to use the CGS algorithm, because doing so would take more

time than using a faster algorithm even if the competing method requires more objective function evaluations, on average, to achieve the same result.

The knapsack problem, traveling salesperson problem (TSP), and the welded beam problem are used to test the computational expense of the CGS algorithm. For each test problem, the CGS algorithm is run with 500 training points (N_{tr}), 500 new points sampled at each iteration (N_s), and a total of 10,000 objective function evaluations. Three measurements are taken at each iteration: training time, classification time, and sampling time. The training time is the time it takes to train the classifier with the 'expensive' points, which are points from the current iteration that have been evaluated with the objective function. The classification time is the time required to use the classifier to classify each 'cheap' point in the set of all candidate solutions that has not yet been evaluated with the expensive objective function. Lastly, the sampling time is the time required to use the outputs from the classifier (classes and posterior probabilities) to sample points from the set of cheap points for subsequent objective function evaluation.

In Figure 4.17, the training time versus the number of training points is shown for the three test problems. In all test cases, the training time increases linearly with the number of training points. This behavior is expected, because the classifier does not become more complex with each new training point. That is, there are no additional terms or parameters in the classifier equations. The test problem with the slowest training time is the knapsack problem, followed by the TSP, and the problem with the fastest training times is the welded beam problem. The problem-specific training time, in this case, is correlated with the number of independent variables for each respective problem (20 for the knapsack problem, 10 for the TSP, and 6 for the welded beam problem). For each independent variable, the training function must work through all of the training points and count the number of times each variable in the training set takes

on a specific discrete value. When a variable is conditionally dependent on one or more other variables (has parent nodes in the Bayesian network), the training function must count the number of times a specific combination of variable and parent variables appear. The connectivity of the Bayesian network has a slight, but relatively inconsequential, impact on the time complexity of this counting process because the values for a particular variable and its parents can be counted simultaneously in a single step.

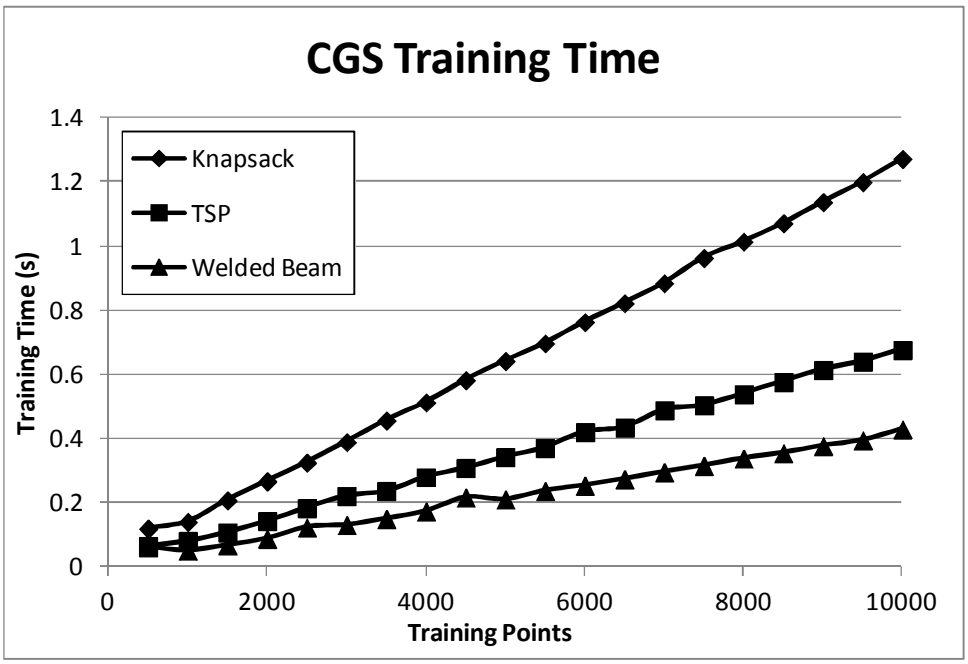


Figure 4.17: Training time versus number of training points

The number of training points has an insignificant effect on the classification and sampling times. However, the number of cheap points has a significant effect on these metrics. The number of cheap points changes very slightly from one iteration to the next, because it decreases by an amount equal to the number points sampled for expensive iteration, and this number is very small compared to the total number of iterations. Therefore, it is not very useful or informative to plot the classification and sampling

times versus the number of cheap points. Instead, the average classification and sampling times of the 20 iterations performed in this time complexity study versus the average numbers of cheap points at each iteration are calculated and plotted, as shown in Figure 4.18. In general, the classification and sampling times increase linearly with the number of cheap points. This trend is expected, because the amount of work performed by the classifier and the sampling step of the CGS algorithm is directly proportional to the number of cheap points that need to be classified and considered for sampling.

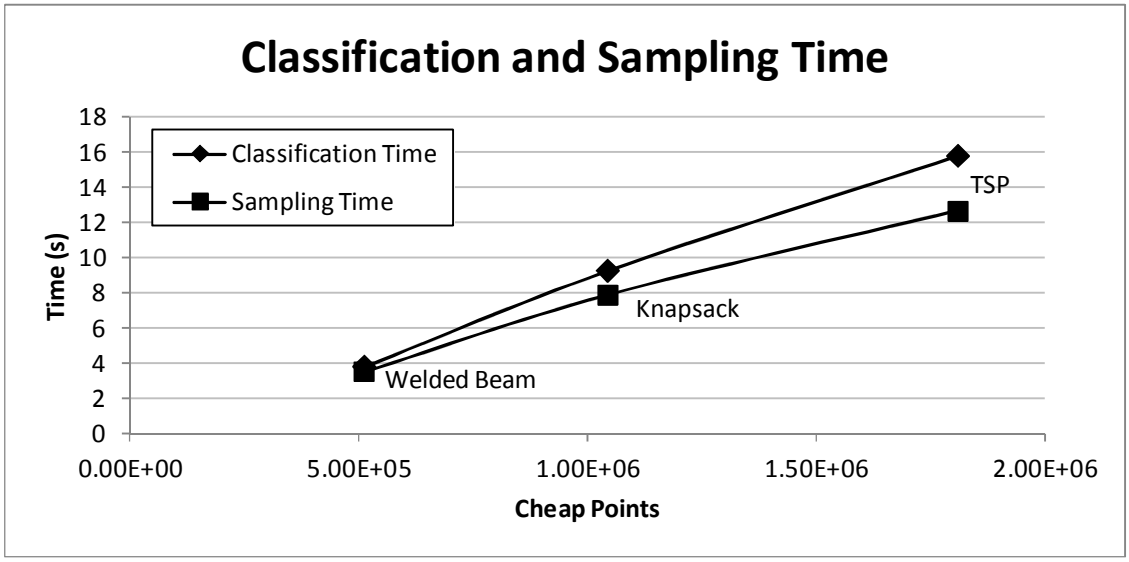


Figure 4.18: Classification time and sampling time versus number of classified points

In Figure 4.19, the total CGS computational time is decomposed into its constituents. Most of the computational expense in the CGS algorithm is associated with the cheap point classification and sampling steps. Training time is a very small percentage of the total time required by the CGS algorithm, while the classification time and sampling time are approximately equal and make up the remaining percentages. Classification and sampling times are large due to the large number of cheap points that must be handled in these steps. For the classification time, all of the cheap points must

be evaluated with the classifier, and this takes a significant amount of time when there are a large number of these points. The sampling step is also somewhat computationally intensive, because all of the cheap points must be checked to determine if they qualify as high-certainty high-class points, high-certainty low-class points, or uncertain points.

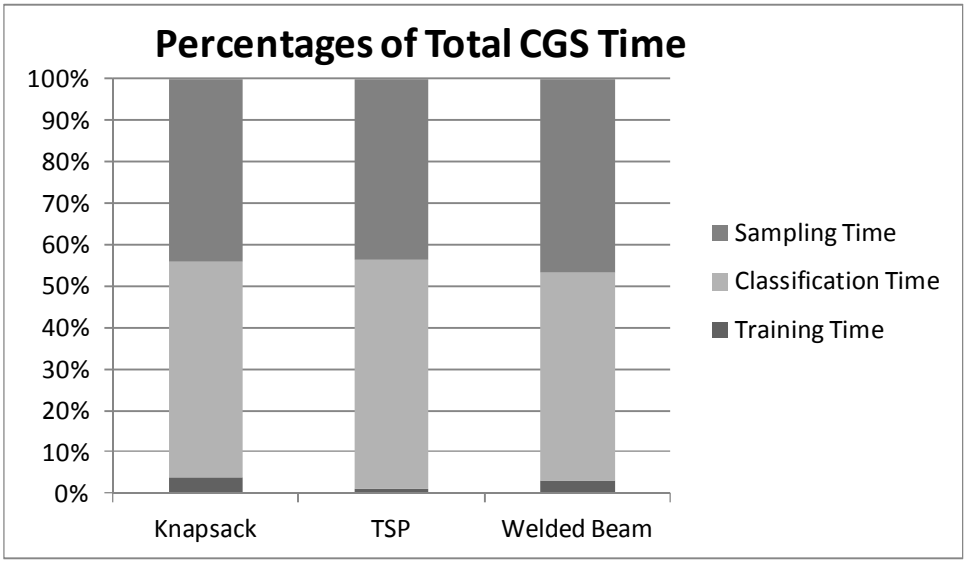


Figure 4.19: Percentages of total CGS time

The accumulated computational expense of the CGS algorithm over all iterations in a single run is highly dependent on the user defined parameter N_s, which is the number of points that are sampled for objective function evaluation at each iteration. When this parameter is smaller, more CGS iterations can be performed for a fixed number of objective function evaluations. Accumulated computation time of the CGS algorithm increases in this case, because the large set of cheap points must be classified and sampled at each iteration. On the other hand, choosing larger N_s values may have adverse effects on the rate of convergence towards optimal solutions. During the first few CGS iterations, the number of training points in the classifier is small, and the classifier may not be providing accurate predictions of categorical solution quality. If N_s is set to a large

value, a large number of points are sampled based on the outputs of an inaccurate classifier. Larger N_s values may result in an inefficient search of the design space, and the total number of function evaluations may be increased. Therefore, smaller values of N_s are generally preferred, especially when the objective function is expensive to evaluate.

The CGS algorithm is more computationally expensive than a typical GA, because a GA's selection, crossover, and mutation operations involve simple manipulations of strings of integers. However, the CGS method converges towards the known global optima of the test problems significantly faster than GAs. If the objective function is very fast and inexpensive, it may require less total time to use a GA even if more objective function evaluations are required. On the other hand, if the objective function is expensive to operate, the benefit of rapid convergence offered by the CGS method outweighs the drawback of the additional computational expense of the CGS algorithm. For a specific problem, there is a specific level of objective function computational expense above which it is worthwhile to use the CGS method over GAs.

Estimations of these values for the three test problems, herein referred to as the "Expensive Evaluation Payoff Times," are provided next. For each test problem, the CGS and GA processes are terminated after a predetermined number of objective function evaluations. On average, the GA does not converge to the known global optima before the maximum number of evaluations are performed. The average best objective function valued that is found with the GA is designated as a "desired" objective function value. Next, the number of function evaluations needed by the CGS method to achieve these desired values is identified. This information is used in conjunction with the results of this time complexity study to estimate the expensive evaluation payoff time for each of these test problems. The number of training points and the number of points sampled at

each iteration are the same as those for the rate of convergence tests in Section 4.3. The results of this process are presented in Table 4.7.

Table 4.7: Expensive evaluation payoff time of the CGS method

Test Problem	*GA Objective Function Evaluations*	*Average Best GA Objective Function Value*	*Evaluations Needed by CGS to Find Equivalent Objective Function Value*	*Expensive Evaluation Payoff Time (s)*
Knapsack	1200	788.34	350	0.10
TSP	8000	344.81	1900	0.08
Welded Beam	2000	$2.0834	450	0.02

For all three test problems, the expensive evaluation payoff time of the objective function is very small (less than or equal to one tenth of a second). That is, if the time required for each objective function evaluation is greater than the expensive evaluations payoff times in Table 4.7, then it takes the CGS algorithm less time to identify a solution that has the preferred objective function evaluations. In other words, even though the CGS algorithm is itself more expensive to run than a GA, the time saved by identifying better solutions with fewer objective function evaluations is significant enough to warrant its use. In this analysis, it is assumed that the GA can be run in zero time. When the actual time of the GA is accounted for, the expensive evaluation payoff time would be even smaller.

4.5 Chapter Summary and Closing Remarks

In this chapter, the classifier-guided sampling (CGS) method for solving computationally expensive, discrete variable, discontinuous design problems is presented. The CGS method uses a Bayesian network classifier to provide categorical estimates of the quality of candidate solutions without requiring an expensive simulation for each test point. The classifier outputs are used to guide the search process towards solutions that

have a high probability of improving on the current best known solution. The method is tested on a set of three discrete variable design problems. Results show that the CGS method significantly improves the rate of convergence towards known optima when compared to GAs and random search.

The success of the CGS method can be attributed to the use of a classifier to serve as an inexpensive approximate of global performance. This approach is similar to direct sampling metamodel-based design optimization techniques, in which a discrete variable classifier is used in place of a continuous variable metamodel to guide the search process. However, the CGS method is capable of solving discontinuous design problems that are inappropriate for traditional metamodel use. To date, GAs have been shown to be effective at solving a vast array of problem types and sizes. However, the CGS method outperforms GAs when applied to the three test problems in this research.

Chapter 5: All-Electric Ship Energy System Modeling

Design of a naval warship is a complex, multidisciplinary engineering problem. Accordingly, one of the objectives of this research is to investigate the effects of sizes, locations, and distribution architectures of energy system components on performance metrics such as fuel consumption and survivability. To perform this task, a dynamic energy system model is needed that is capable of rapidly analyzing a large variety of configurations.

In this chapter, a highly reconfigurable system-level model that enables performance assessment of total-ship energy distribution system configurations is developed. As an input, the model accepts a candidate design in the form of a string of discrete variables to represent the locations, sizes, and connectivity of energy system components. Time-dependent load profiles and damage scenarios are used as inputs to capture the dynamic nature of notional real-life situations. Metamodels are used to provide fast approximations of select subsystem component models. The modeling approach presented here supports the use of discrete variable optimization algorithms, such as the classifier-guided sampling method, to perform rapid design and performance analysis of candidate system designs.

5.1 System-Level Modeling

Significant effort has been dedicated to system-level modeling of electric ship thermal and electrical distribution systems. The Dynamic Thermal Modeling and Simulation Framework (DTMS) [21-23], developed at Applied Research Laboratories at UT Austin, is a tool for modeling and simulating system-level dynamic thermal systems. The Virtual Test Bed (VTB) was developed at the University of South Carolina for

supporting “design, analysis, and virtual prototyping of electric power systems” [24]. VTB has since been updated to include thermal systems [25].

The reconfigurable total-ship energy system model presented here differs from DTMS and VTB in that the user can rapidly test the performance of a large variety of system architectures by manipulating vectors of discrete variables. This property of the system model is especially useful when an automated design optimization algorithm, such as the classifier-guided sampling method, is used to search for configurations that meet specific objectives such as energy efficiency or survivability. The tradeoff is that the modeling approach lacks some of the detailed physical phenomena that are present in other modeling approaches. However, the intended use of the framework is for early-stage ship design, when a designer must perform initial studies to estimate the overall efficiency, survivability, and flexibility of a large set of candidate designs.

Total-ship energy management involves several flowing commodities that are distributed throughout the ship to provide electric power and waste heat removal capacity where it is needed. These include chilled water, freshwater, seawater, refrigerated air, and electric power. Heat is removed from chilled water with compressor driven chilled water plants. Fresh water rejects heat directly to seawater through heat exchangers. Refrigerated air, which cools compartments with equipment and personnel heat loads, is produced by passing circulated air over cooling coils that are cooled with water from the chilled water system. Much of the increased heat loads on the future all-electric ship are expected to be rejected into the HVAC system or directly into the chilled water system [110]. Therefore, the fresh water distribution system is not included in the model presented here.

The electrical, chilled water, and refrigerated air systems are modeled as a hierarchy in which each child layer is dependent on its parent to function, as shown in

Figure 5.1. It is necessary to model all three layers because the distributed thermal and electrical commodities have system-level interdependencies that must be captured by the modeling framework. For example, electric power is required for the generation and distribution of chilled water, and chilled water and electric power are required for the generation of refrigerated air through heat exchangers and distribution fans, respectively.

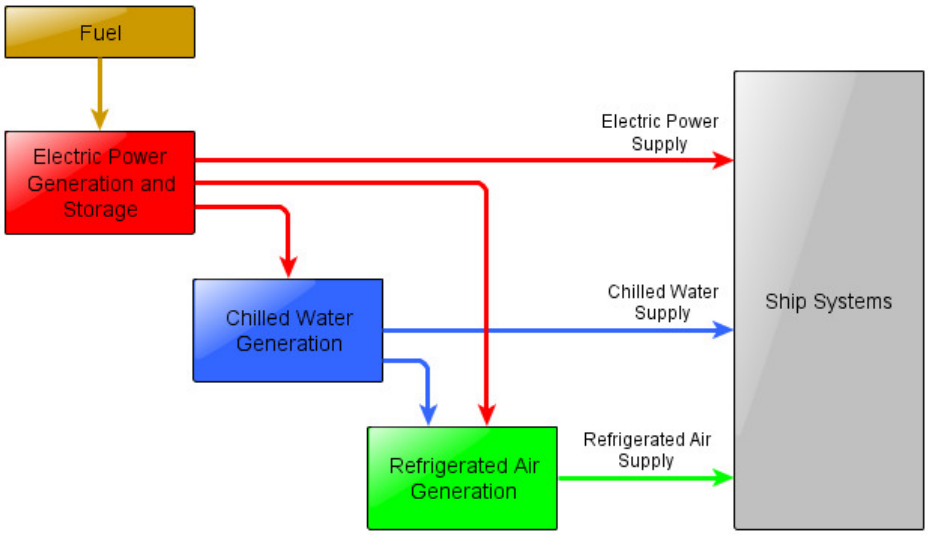


Figure 5.1: Interdependence of distributed energy system layers

The operating level of one layer may affect demand of another layer. For example, electrical loads may require chilled water to operate at safe operating temperatures. Therefore, chilled water and refrigerated air demand may be proportional to or dependent on electric power consumption.

Each energy distribution layer has its own zonal distribution architecture, as shown in Figure 5.2. Studies have shown that zonal electrical distribution architectures result in systems with lower acquisition costs, lower weight, and better operational flexibility [8]. The benefits of a zonal architecture are not limited to electrical distribution systems. For example, Shiffler [9] estimated that a zonal architecture for a

shipboard fire main system resulted in cost and weight savings of 20% and 9%, respectively. Zonal electrical distribution architectures have been implemented in several classes of US Navy warships, including DDG-51 guided missile destroyers, LHD-8 assault ships, and LPD-17 transport vessels [10].

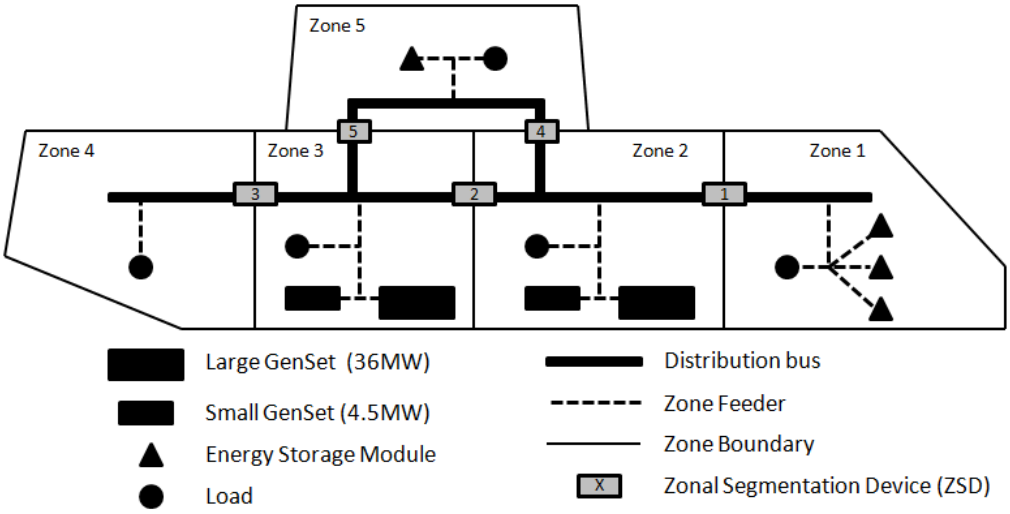


Figure 5.2: Zonal energy distribution architecture

In Figure 5.2, the longitudinal distribution bus connects the zones along the length of the vessel, and the zone feeders connect loads and generation components to the distribution bus. Zonal segmentation devices (ZSDs) are placed at zone boundaries and represent valves or switches that can be opened or closed to isolate a zone or set of zones from others. A generation component in one zone can supply distributed commodities to loads in other zones through the distribution bus. Should a particular zone become disconnected from all generation components, storage components can be made available until power is restored to the area.

Using the zonal modeling approach, the design and configuration variables are represented as vectors and matrices of discrete variables, as is discussed in Chapter 1.

For example, in Figure 5.2, a notional electrical distribution system is shown. There are two 4.5MW and two 36MW generators. One of each type of generator is placed in zones 2 and 3. There are three electrical energy storage modules in zone 1, and one storage module in zone 5. Therefore, the matrices that represent the configuration in Figure 5.2 are given by

$$\begin{aligned} Gens &= \begin{bmatrix} 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \end{bmatrix} \\ Storage &= \begin{bmatrix} 3 & 0 & 0 & 0 & 1 \end{bmatrix} \end{aligned} \tag{5.1}$$

where *Gens* is a matrix whose first row represents the quantity of 4.5 MW generators and second row represents the quantity of 36 MW generators in each zone, and *Storage* is a vector used to represent the number of storage modules in each zone. The states of the ZSDs are represented with a vector of binary variables. For example, consider a scenario in which it is more efficient for the generators in zone 2 to serve only zones 1 and 2 and for the generators in zone 3 to serve zones 3, 4, and 5. In this case, the states of the ZSDs could be set to

$$ZSD = \begin{bmatrix} 1 & 0 & 1 & 0 & 1 \end{bmatrix} \tag{5.2}$$

where a value of 0 indicates the OFF position and a value of 1 indicates the ON position. In this case, ZSDs 2 and 4 are in the OFF position, meaning that no electric power is able to flow across these zone boundaries.

The description above is intended to provide an overview of the three energy system layers and the manner in which a particular configuration can be represented using vectors of discrete variables. Although the example in this section focuses on the energy power generation and storage layer, the zonal distribution architecture for each of the other two layers in Figure 5.1 can be modeled analogously. In the following sections,

detailed discussions of the electrical, chilled water, and refrigerated air subsystem models are provided.

5.2 ELECTRICAL SUBSYSTEM

The inputs and outputs for the zonal electrical system model are shown in Figure 5.3. Three sets of design variables define the electrical system configuration: generators, storage modules, and distribution bus ZSD states. Each energy storage module has two user-defined properties: energy capacity and a power capacity. The user is free to define these characteristics to match those of any size flywheel, battery, or other energy storage device. The user-defined electric power demand is given to the model as a set of time-dependent load profiles, where each zone has its own unique load profile to represent the electric power demand of electrical components that exist in each zone of the ship. The electric power needed for chilled water and refrigerated air production are outputs of the chilled water and refrigerated air models (to be described in following sections).

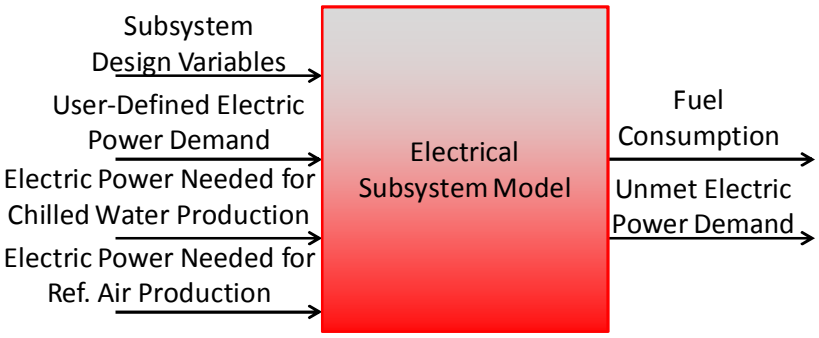


Figure 5.3: Electrical system model inputs and outputs

The outputs of the electrical system model are used to assess the performance of candidate ship configurations. Fuel consumption is used to assess the operating efficiency of a design, and unmet electric power demand is used to evaluate survivability and operating flexibility.

In Figure 5.4, a block diagram of the electrical subsystem is shown to illustrate the flows of information through the model. On the left hand side, the subsystem model inputs are shown. The first three inputs (Zonal Generator Power Capacity, Zonal Storage Power and Energy, and ZSD Topology) are directly inferred from the design variable values. The final input (Total Zonal Electric Power Demand) is the sum of all electric power demand, including the user-defined power demand, and the electric power needed for chilled water and refrigerated air production. The outputs shown on the right hand side of Figure 5.4 match those shown in Figure 5.3.

The intermediate blocks in the figure perform five main functions: calculate the connectivity of the system components, configure the available generators, determine generator and storage operating loads, track storage state of charge, and calculate fuel consumption. The blocks in Figure 5.4 are explained in more detail in the following sections.

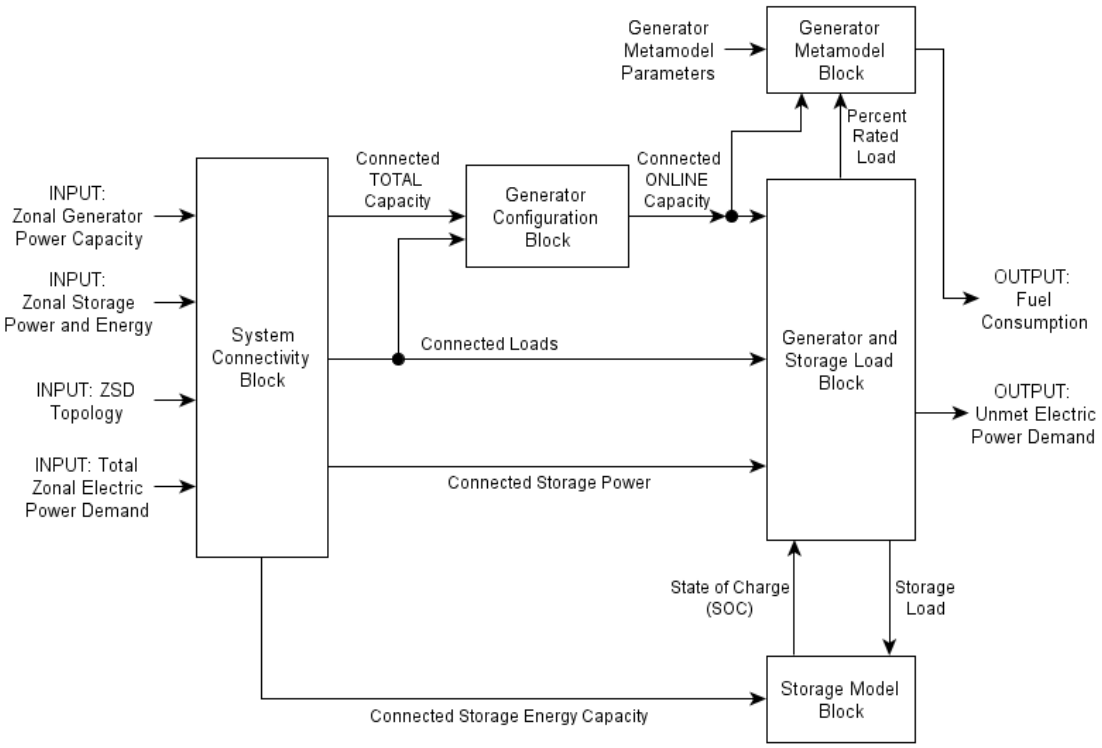


Figure 5.4: Electrical subsystem model block diagram

5.2.1 System Connectivity Block

The System Connectivity Block uses the input vector of ZSD positions to define and calculate important characteristics related to the connectivity of the electrical distribution system. This task is achieved with the use of a connectivity matrix, *CON*. *CON* is a symmetric, binary matrix in which the element at the intersection of the i_{th} row and j_{th} column indicates whether zones i and j are connected. For example, consider the ZSD positions used in the example in Section 5.1:

$$ZSD = \begin{bmatrix} 1 & 0 & 1 & 0 & 1 \end{bmatrix} \tag{5.3}$$

This ZSD topology results in zones 1 and 2 being connected to each other, and they are separated from Zones 3, 4, and 5 which are also connected to each other. Therefore, the *CON* matrix that results from this topology is given by

$$CON = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 \end{bmatrix} \tag{5.4}$$

The *CON* matrix is used to transform vectors representing *zonal* system properties into vectors that represent *connected* system properties. *Zonal* system property vectors are five element vectors that represent the total generation capacity, loads, and/or storage capacity for each zone. However, *connected* system property vectors represent the total capacity, loads, and/or storage energy and power that is connected to and available for use by each zone. For example, the zonal capacity of a system with a 36 MW generators in zones 2 and 3 and 4.5 MW generators in zones 3 and 4 is given by:

$$zonal\_cap = \begin{bmatrix} 0 & 36 & 40.5 & 4.5 & 0 \end{bmatrix} (\text{MW}) \tag{5.5}$$

That is, there is 36 MW of electric power generation capacity in Zones 2, 40.5 MW of generation capacity in Zone 3, and 4.5 MW of generation capacity in Zone 4. Using the ZSD vector in Equation (5.3), the connected capacity is obtained by multiplying the zonal capacity vector with the *CON* matrix:

$$connected\_cap = zonal\_cap * CON = \begin{bmatrix} 36 & 36 & 45 & 45 & 45 \end{bmatrix} (\text{MW}) \tag{5.6}$$

The same calculation is also used to transform the loads, storage energy capacity, and storage power capacity from zonal system properties to connected system properties.

Representing the system properties with *connected* vectors rather than *zonal* vectors enables the model to treat a set of connected zones as a single zone with all connected generators, storage modules, and loads in one lumped sum.

5.2.2 Generator Configuration Block

The function of the Generator Configuration Block is to decide which of the available generators to turn on to meet current electric power demand. In many cases, it is advantageous to operate fewer than the maximum number of available generators, because doing so consumes less fuel. To illustrate this point, the performance curves for the RR4500 (4.5 MW) and MT30 (36 MW) gas turbine generator sets are shown in Figure 5.4.

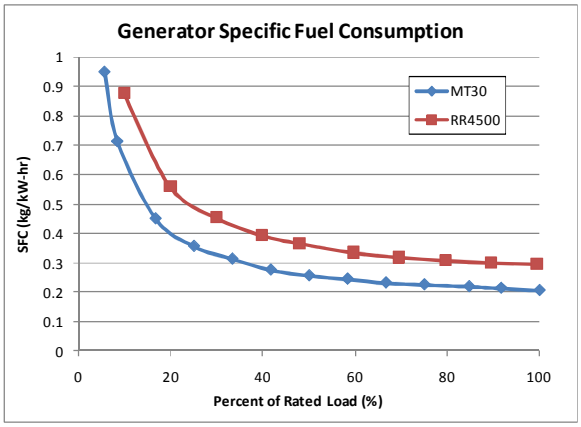


Figure 5.4: Gas turbine generator specific fuel consumption versus percent of rated load [13]

Figure 5.4 shows that the specific fuel consumption (kg/kW-hr) of the generators decrease with increasing percent of rated load. Therefore, the most fuel efficient generator configuration is the one that runs as few generators as possible while providing sufficient capacity to meet electric power demand. Standard naval engineering practice

dictates that any online connected generators must be operated at the same percent of their rated load [13].

A rule-based strategy is employed to ensure that a generator is online only if it is needed to meet demand. For example, consider a scenario in which Zones 3, 4, and 5 are isolated from Zones 1 and 2 using the electrical segmentation devices and there is one RR4500 and one MT30 in Zone 3. There is also one RR4500 and one MT30 in Zone 2. If the total electric power demand of Zones 3-5 is 20 MW, only the MT 30 will be online because the RR4500 is not needed and running it would result in less efficient operation. If, at the same time, the load in Zones 1 and 2 is 40 MW, both of the generators in Zone 2 would be online and operating at the same percent load because the MT30 would not be able to produce the required electric power on its own.

An undesirable characteristic of gas turbine generators is that they do not respond immediately to sudden load increases. In addition to determining the number of available generators to run, the Generator Configuration Block has a feature that gives the user the option to simulate this effect. This task is achieved with a dynamic rate limiter, which places a user-specified upper limit on the rate at which the connected online generation capacity can increase. The generators are assumed to be in standby mode at all times, meaning that they are warm and available for operation at a moment's notice. That is, the additional delays that are normally associated with a "cold" startup are not accounted for in this model.

5.2.3 Storage Model Block

The purpose of the Storage Model Block is to track the state of charge (SOC) of connected storage module capacity at each time step. Energy storage modules are included in the total-ship energy system model to provide temporary additional electric

power when the gas turbine generators are unable to meet demand. This situation may arise during sudden increases in electric power demand while the generators are ramping up, or when total demand exceeds the total capacity of the gas turbine generators. Storage may also be useful during a damage scenario if a generator becomes unavailable, or a particular zone is completely disconnected from generators.

The SOC is used by the Generator and Storage Load Block to inform decisions about whether to charge the storage modules with excess generating capacity or discharge the storage modules to meet demand that cannot be met by the generators. To perform this function, the storage model requires two flows of information. First, it must have knowledge of the connected energy capacity of the storage modules. SOC is the ratio of stored energy in the storage modules to total energy capacity of the storage modules. Therefore, the connected energy capacity of the system is required to make this calculation. The second flow of information required by the Storage Model Block is the storage load, which is measured in energy per unit time. The storage load is positive if the storage modules are being charged by the generators and it is negative if they are discharging to supply additional power to the system. The storage model integrates the storage load to track total energy flowing in and out of the storage modules. At each time step of the solution process, the quantity of stored energy in the storage modules is divided by the total energy capacity of the storage modules. The result is connected SOC, which is passed to the Generator and Storage Load Block.

5.2.4 Generator and Storage Load Block

The Generator and Storage Load Block performs three interrelated functions. First, it serves as a controller for the storage modules; it makes a decision about whether to charge, discharge, or do nothing. Second, this block calculates the percent of rated

capacity at which each generator is operating. Lastly, the Generator and Storage Load block calculates the amount of electric power demand that goes unmet as a result of inadequate generation and storage power capacity.

The task of controlling the storage modules requires knowledge of the SOCs of the storage modules (provided by the Storage Model Block), the power capacity of the storage modules, the electrical loads, and the online generator capacity. There are also two constant parameters, the maximum and minimum allowed states of charge of the storage modules (SOC_{max} and SOC_{min}), which are specified by the user.

The following rule-based control strategy is employed to determine the operating mode of the storage modules. If there is excess online generator capacity and the current SOC is less than SOC_{max}, the excess generator capacity is used to charge the storage modules. Likewise, if electric power demand exists that cannot be met with the generators, and the SOC of the storage modules is greater than SOC_{min}, the storage modules are discharged to assist the generators with meeting unmet demand. If neither of these conditions exist, storage is not charged or discharged, generators meet loads without the assistance of storage modules, and they do not charge storage modules. If electric power demand exceeds available generator capacity and the current SOC is less than or equal to SOC_{min}, the excess demand remains unmet and is given as a model output.

The storage operating mode must be determined before the other outputs of this block can be calculated. The percent of rated load of online generators cannot be calculated unless it is known whether the generators are charging the storage modules in addition to meeting shipboard electrical loads, because total generator output is equal to the electrical loads plus additional demand that is associate with charging the storage modules. Once the charging mode is known, the percent load is calculated as the ratio of

total generator output to connected online generator capacity. The percent load is passed to the Generator Metamodel Block to calculate the fuel consumption of the generators. Lastly, unmet demand is calculated as the total electric power demand minus generator and storage output. Therefore, it cannot be calculated until it is known whether storage is available and operating to assist the generators with meeting excess electric power loads.

5.2.5 Generator Metamodel Block

The Generator Metamodel Block is used to calculate the rate of fuel consumption using the percent of rated load at which each generator is operating. To achieve this task, the block requires the percent of rated load and the generator metamodel parameters. In addition, the Generator Metamodel Block also requires knowledge of which generators are actually online and running. Therefore, the connected online capacity is also passed to this block to deduce which of the 4 generators are online at each time step.

In Figure 5.4, specific fuel consumption (SFC) curves are shown for the RR4500 and MT30 generators. These curves are generated in Microsoft Excel using a table of data points provided by Syntek [13]. In the Generator Metamodel Block, metamodels are used to interpolate these data points to provide estimates of SFC versus the percent of rated load of the two generators. In the metamodeling scalability and comparison studies in Chapter 3, kriging is shown to achieve good prediction accuracy with few training points for all six test problem. Unlike radial basis functions (RBF) and support vector regression (SVR), the performance of kriging is significantly less sensitive to user-defined tuning parameters. Therefore, kriging metamodels are used to approximate the specific fuel consumption curves of the generators in the Generator Metamodel Block. In Figures 5.5-5.6, the kriging metamodels and their training points are shown. SFC

decreases with increasing percent of rated load for both generators, and the average SFC for the MT30 is slightly lower than that of the RR4500.

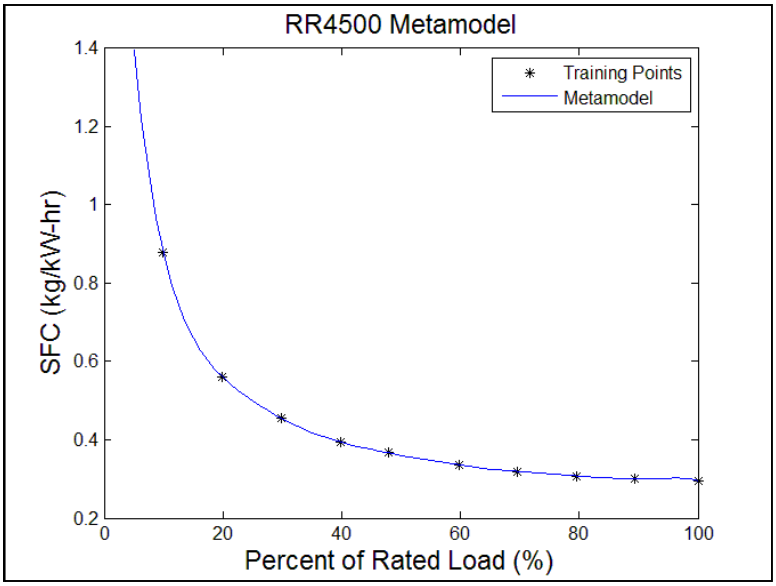


Figure 5.5: RR4500 metamodel and training points

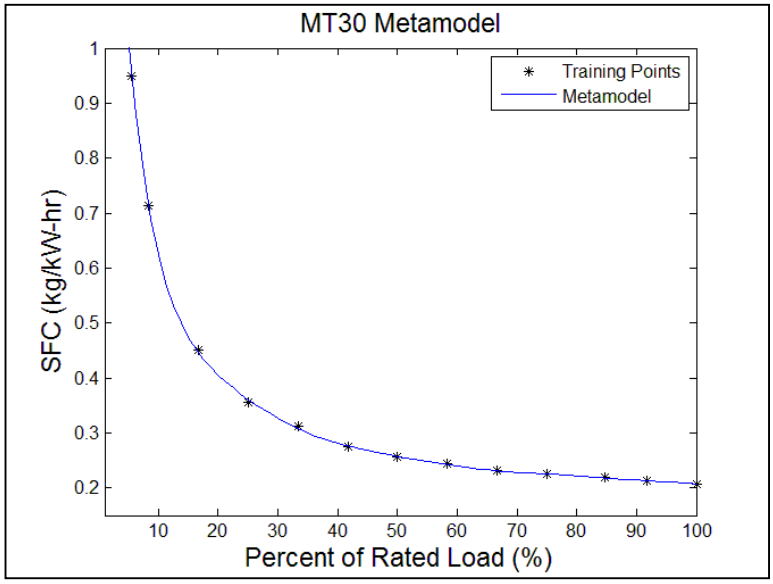


Figure 5.6: MT30 metamodel and training points

5.3 CHILLED WATER SUBSYSTEM

The zonal chilled water (CW) system inputs and outputs are shown in Figure 5.7. For the CW system, the design variables include the number of CW plants in each zone and the states of the distribution bus ZSDs. The user-defined CW demand is given to the model as a time-dependent load profile, and each zone has its own unique load profile to represent the CW capacity (in kW) that is needed to remove waste heat from shipboard systems and components. The CW needed for refrigerated air production is an output of the refrigerated air subsystem model. This load is expected to be relatively large because all of the refrigerated air is produced by forcing it over CW cooling coils.

There are two outputs of the CW subsystem model. The first output (Electric Power Needed for Chilled Water Production) is fed back to the electrical subsystem model and is added to the user-defined electric power demand profile. The second output (Unmet Chilled Water Demand) is used to assess performance metrics such as survivability of a candidate ship design.

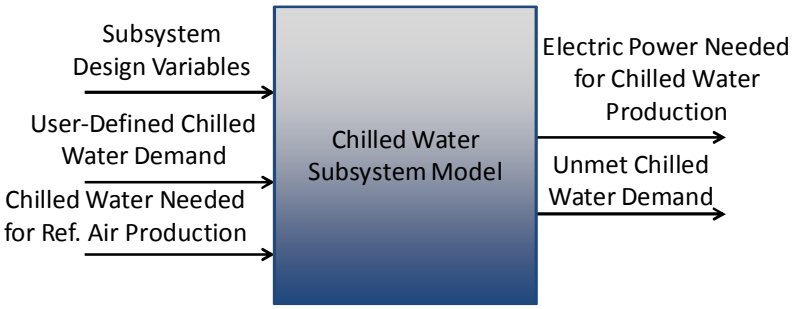


Figure 5.7: Chilled water subsystem model inputs and outputs

In Figure 5.8, A block diagram of the CW subsystem is shown. Contrary to the electric power subsystem model, there is no storage capability for the CW subsystem. Therefore, there is no storage model block and no storage controller. The subsystem model inputs are on the left hand side of the figure. The first two inputs (Zonal Chilled

Water Plant Capacity and Chilled Water ZSD Topology) are directly inferred from the design variable values. The final input (Total Zonal Chilled Water Demand) is the sum of all CW demand, including the user-defined CW demand, and the CW needed for refrigerated air production. The outputs shown on the right hand side of Figure 5.8 match those shown in Figure 5.7.

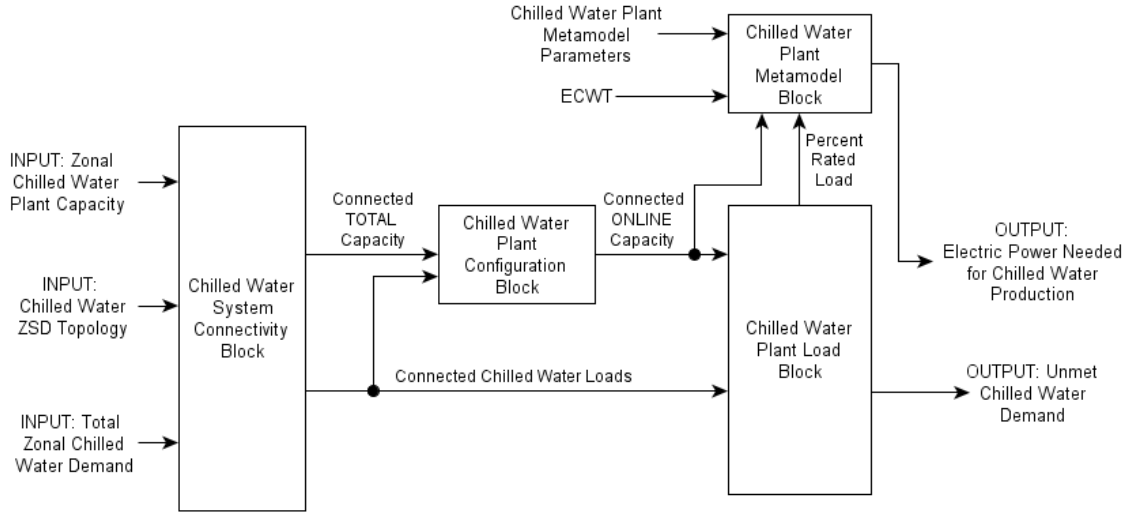


Figure 5.8: Chilled water subsystem model block diagram

There are four primary functions performed within the CW subsystem: determine CW subsystem component connectivity, configure available CW plants, determine CW plant operating levels, and calculate the electric power demand of the CW plants. These functions are explained in more detail in the following sections.

5.3.1 Chilled Water System Connectivity Block

The Chilled Water System Connectivity Block is very similar to the system connectivity block featured in the electrical subsystem model. Like the connectivity block in the electrical subsystem, this block uses the CW ZSD positions to create a connectivity matrix. The connected CW loads and the connected total CW plant capacity

are calculated by multiplying the zonal CW loads and zonal CW plant capacity vectors with the connectivity matrix.

5.3.2 Chilled Water Plant Configuration Block

The purpose of the Chilled Water Plant Configuration Block is to decide which of the available CW plants to run to meet chilled water cooling demand. The general trend for the CW plant electric power consumption is very similar to that of fuel consumption by the electric power generators; electric power demand per ton of refrigeration output decreases with increasing percent of rated load (1 ton of refrigeration = ~3.5 kW). This trend is shown in Figure 5.9. The entering condenser water temperature (ECWT) also has a significant effect on CW plant efficiency; lower ECWT results in lower electric power demand by the CW plant. The temperature of the seawater is used for the ECWT input because seawater is usually used to remove heat from the condenser, and the ECWT is a user-defined constant in the CW subsystem model.

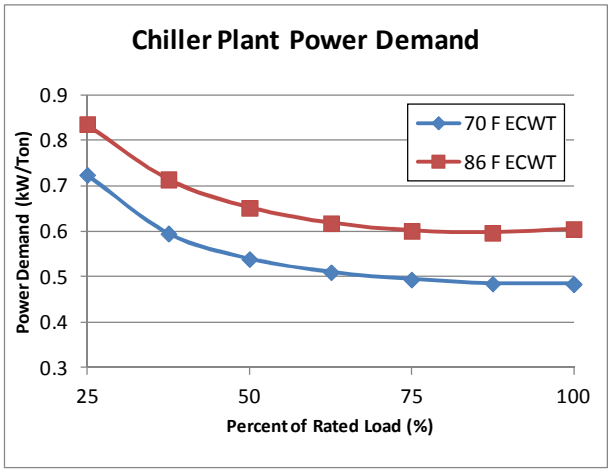


Figure 5.9: Chiller plant power demand versus percent of rated load [14]

Figure 5.9 shows that the CW plants are most efficient when operating near 100% of their rated load. Therefore, it is advantageous to run as few CW plants as possible at

all times. In the Chilled Water Plant Configuration Block, a rule-based strategy is used to ensure that a CW plant is only online if it is needed to meet current CW demand.

5.3.3 Chilled Water Plant Load Block

The Chilled Water Plant Load block performs two functions. First, it calculates the percent of rated load of all connected online CW plants. It performs this task by dividing the connected CW demands by the connected online CW plant capacities. Second, this block calculates any unmet demand that exists as a result of large loads that exceed the available CW plant capacity. The unmet CW demand is an output of the CW subsystem, and the percent of rated load is passed to the CW Plant Metamodel block to calculate the electric power demand of the online CW plants.

5.3.4 Chilled Water Plant Metamodel Block

To calculate the electric power needed for CW production, a chiller metamodel is developed that uses percent of rated CW plant capacity and entering condenser water temperature (ECWT) as inputs. The percent of rated load is provided by the Chilled Water Plant Load Block, and the ECWT is a user-defined constant. In addition to these two parameters, this block requires knowledge of which chilled water plants are online and running. Therefore, the connected online CW plant capacity is also passed to this block to deduce which of the chilled water plants are online at each time step.

In Figure 5.9, the electric power demand curves are shown for the CW plant. These curves were generated using Microsoft Excel using a table of data for centrifugal compressor CW plants [14]. In the Chilled Water Plant Metamodel Block, a kriging metamodel is used to interpolate these data points to provide estimates of the CW plant electric power demand versus the ECWT and the percent of rated load of the CW plants. Kriging is used in this case for the same reasons that it is used to approximate generator

specific fuel consumption curves in the electrical subsystem. In Figure 5.10, the two-dimensional kriging model surface and its training points are shown.

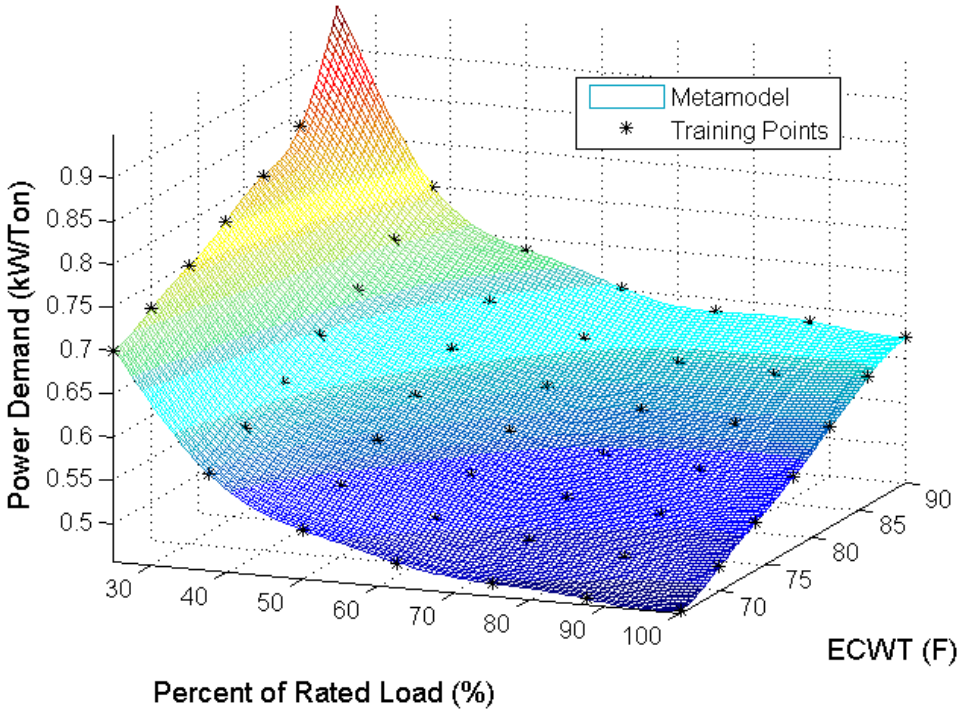


Figure 5.10: Chilled water plant metamodel and training points

5.4 REFRIGERATED AIR SYSTEM

The inputs and outputs of the zonal refrigerated air (RA) system are shown in Figure 5.11. The inputs of the RA subsystem model are the design variables and the user-defined RA demand. The outputs of the system are the electric power and CW needed for RA production, and the unmet RA demand.

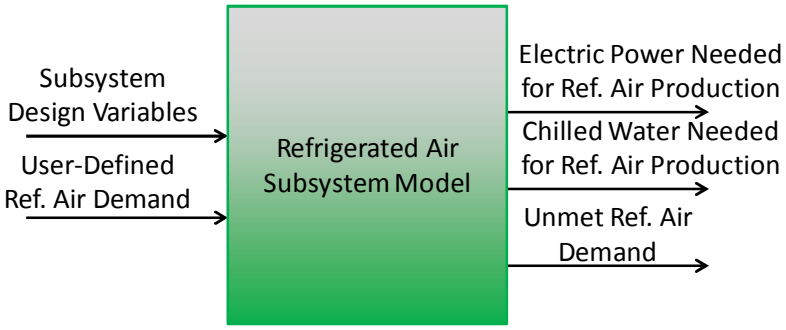


Figure 5.11: Refrigerated air system inputs and outputs

The only design variables for the RA system are quantities of cooling coils in each zone. The cooling coils are air/water heat exchangers in which circulated cabin air is passed over fins that are cooled using water from the CW system. Unlike the electric power system and the CW system, there are no lengthwise busses or zonal segmentation devices for the RA system; RA that is produced in one zone is not transported to meet demand in an alternate zone. However, it is assumed that all RA produced within any given zone can be distributed to remove waste heat throughout that entire zone. The reason for this restriction on the RA subsystem model is that zone boundaries are usually aligned with watertight bulkheads [10], and passing large air ducts across these boundaries would compromise the effectiveness of this architectural feature should one of the zones become flooded. Furthermore, such a duct would be large compared to an electrical or CW bus to accommodate potentially large air flow rates. For the reasons described above, all required RA for a particular zone is to be generated in that zone.

Like the CW and electric power user-defined demands, the user-defined RA demand is given to the model as a time-dependent power load profile (in kW). Each zone has its own unique load profile to represent the RA capacity that is needed to remove waste heat from components in each zone of the ship.

The block diagram for the RA subsystem is shown in Figure 5.12. The RA subsystem is significantly simpler than those of the CW and electric power subsystems for two reasons. First, there are no storage modules to be modeled and controlled. Second, there is no system-wide duct work or ZSDs for the RA system. Therefore, there is no need to transform zonal RA subsystem properties into connected subsystem properties, as was necessary for the CW and electric power subsystems.

The main blocks of the RA subsystem model serve three functions: configure the available cooling coils, calculate fan electric power demand, and calculate pump electric power demand. These three blocks are explained in more detail in the following subsections.

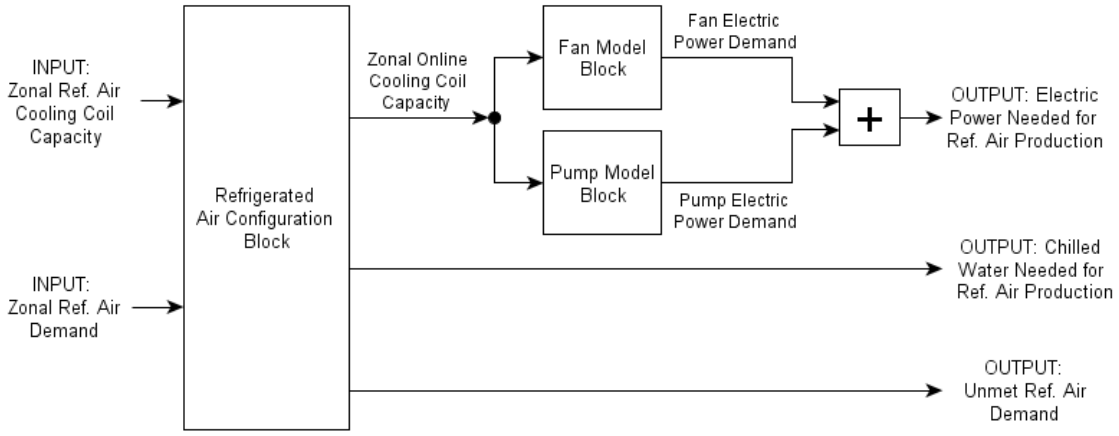


Figure 5.12: Refrigerated air subsystem model block diagram

5.4.1 Refrigerated Air Configuration Block

The purpose of the Refrigerated Air Configuration Block is to determine the number of cooling coils that are online at each time step. The cooling coils either operate at full capacity or are completely off. They do not operate at a partial percentage of their rated capacity, and it is advantageous to operate as few of the available cooling coils as

possible at any given time. Therefore, the Refrigerated Air Configuration Block sets the number of online cooling coils to meet in-zone demand at each time step.

The Refrigerated Air Configuration Block give two outputs in addition to the online cooling coil capacity. First, this block provides the CW capacity needed for RA production, which is equal to the total rated capacity of all online RA cooling coils. Second, this block calculates the unmet RA demand, which is equal to the excess demand in each zone that is unable to be met with the available cooling coils.

5.4.2 Fan and Pump Model Blocks

The source of the cooling capacity of the refrigerated air system is water from the CW system, as opposed to standard household air conditioning units that run on electric power and refrigerant in a vapor compression cycle. However, the RA system requires electric power to operate fans and pumps to overcome pressure losses in piping and cooling coil heat exchangers. Therefore, the purpose of the Fan and Pump Model Blocks is to estimate fan and pump electric power demand. The outputs of these two blocks are summed to provide the electric power demand needed for RA production.

In the Fan and Pump Model Blocks, the fan and pump power is estimated using assumed fan and pump efficiencies, air and CW flow rates, and pressure drops across the coils of the air and water. Specifications for the cooling coil used in the RA subsystem model are given in Table 5.1.

Table 5.1: 60 Series cooling coil parameters, size 61 [110]

Parameters	*Units*	*Values*
Cooling Coil Capacity	Ton/coil	0.8
Air Side Flow Rate	(ft^3/min.)	280
Air Side Head Loss	in. H_2O	0.76
Water Side Flow Rate	gpm/Ton	3.6
Water Side Pressure Drop	ft. H_2O	0.15

The zonal electric power consumption of the RA fans, P_f, is proportional to the number of online cooling coils in each zone and is given by Eq. (5.6):

$$P_f = \frac{N_{cc} Q_a p_a}{\eta_f} \tag{5.6}$$

where N_{cc} is the number online cooling coils, Q_a is the volumetric flow rate of air for each cooling coil, p_a is the air side pressure drop across each cooling coil, and η_f is the fan efficiency. The fans are assumed to have a constant average efficiency of 75% [110].

The water pumps are used to transport the CW both within zones and between zones, and friction losses in intra-zone and inter-zone CW pipes increase the required pumping power. In table 5.2, the length of pipe within each zone and between adjacent zones is given, where the intersection of columns i and j represents the length of pipe between those adjacent zones. If $i = j$, the element in Table 5.2 represents the length of pipe within that respective zone. The assumed head loss for the CW piping is 4 ft. H_2O per 100 ft. of pipe length.

Table 5.2: Chilled water piping distance matrix (ft.)

	Zone 1	*Zone 2*	*Zone 3*	*Zone 4*	*Zone 5*
Zone 1	300	450	-	-	-
Zone 2	450	300	300	-	350
Zone 3	-	300	300	450	350
Zone 4	-	-	450	300	-
Zone 5	-	350	350	-	300

The connectivity of the chilled water piping system affects the required pumping power. For example, if Zones 1 and 2 are connected, there is (300 + 300 + 450) = 1050 ft. of pipe for the chilled water to travel through in this connected set of zones. However, if Zones 1 and 2 are completely isolated, the total length of pipe for water to travel through for these two zones is only (300 + 300) = 600 ft.

Similar to the fan electric power demand, the electric power requirements of the pumps are proportional to the number of online cooling coils in each zone, as each coil adds to the pressure drop that must be overcome with pump effort. The electric power consumption, P_p, of pumps in a group of zones that are connected with the CW distribution bus is given by Eq. (5.7):

$$P_p = \frac{N_{cc} Q_w \left(p_w + p_{piping} \right)}{\eta_p} \tag{5.7}$$

where N_{cc} is the number online cooling coils, Q_w is the volumetric flow rate of water through each cooling coil, p_w is the water side pressure drop across each cooling coil, and η_p is the pump efficiency. The term p_{piping} is the total pipe friction loss that exists among a group of connected zones. The pumps are assumed to have a constant average efficiency of 60%.

5.5 CHAPTER SUMMARY

The three energy subsystems described in the previous sections are strategically connected to form the total-ship energy system model, shown in Figure 5.13. The main inputs, which enter the system-level model on the left hand side of Figure 5.13, are the subsystem design variables and user-defined load profiles. The four outputs, which exit the model on the right hand side of the figure, are the fuel consumption and unmet demands for each subsystem.

The outputs of the electrical subsystem (fuel consumption and unmet electric power demand) flow directly out of the total-ship system model, as they are not used as inputs for the other subsystem models. However, total electric power demand depends on the operating levels and number of online CW plants and RA cooling coils. Therefore, the electric power needed for CW and RA production exit the CW and RA

subsystem models, respectively, and are fed back to the electrical subsystem model after being summed with the user-defined electrical load profile. Likewise, the total CW demand depends on the online RA cooling coil capacity. Therefore, the CW needed for RA production exits the RA subsystem model and is fed back to the CW subsystem model, where it is summed with the user-defined CW load profile.

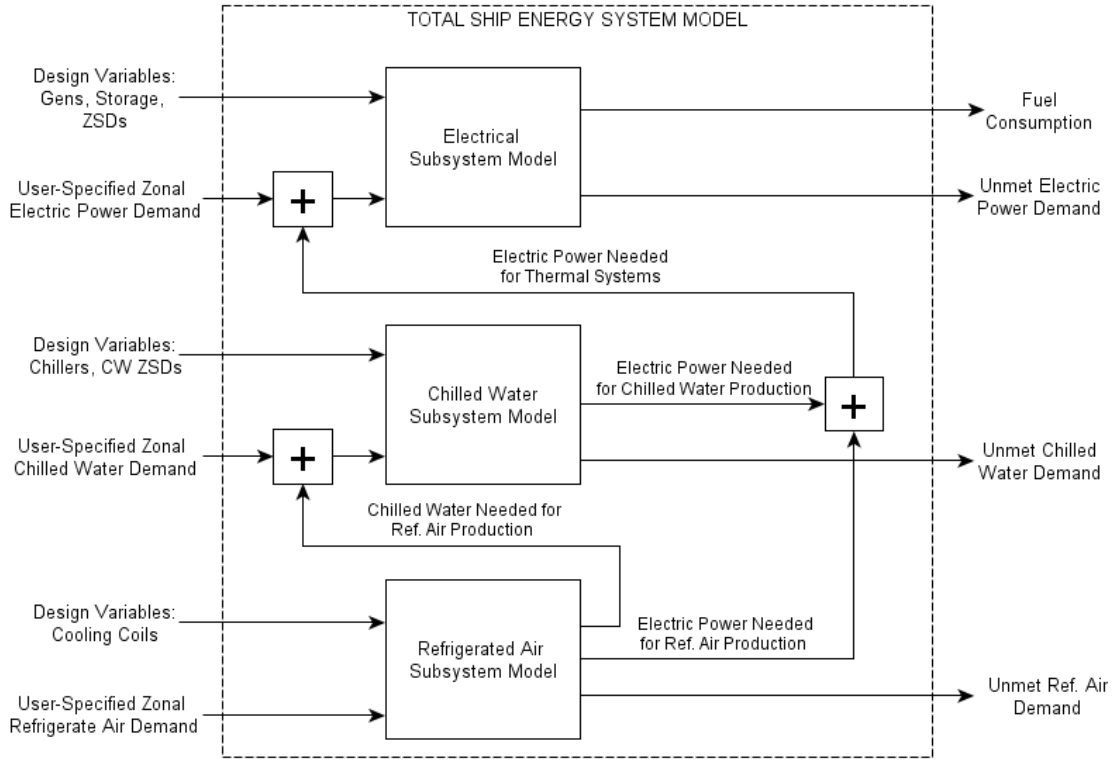


Figure 5.13: Total-ship energy system model

The total-ship energy system modeling approach described in this chapter is a highly reconfigurable, searchable model with discrete variable inputs that can be easily manipulated. Many commercially available modeling platforms do not share this property, and analysis of a new system architecture often requires an entirely new model to be constructed with these platforms. The modeling approach presented here is best suited for early-stage ship architecture design studies in which the engineer desires to

gain a better understanding of how the selection, placement, and configuration of electrical and thermal components affect important performance metrics such as fuel consumption and survivability.

Chapter 6: All-Electric Ship Design Studies

In this chapter, the classifier-guided sampling (CGS) method is used to identify all-electric ship (AES) energy system configurations that consume less fuel and are more survivable. To achieve this task, load profiles are developed and used as user-defined inputs to the total-ship energy system model in Chapter 5. For the fuel consumption study, average annual fuel consumption is calculated by using loads that represent a full spectrum of AES operating modes. The loads are weighted by the anticipated time spent in each operating mode. Notional damage scenarios are imposed on the model to assess the effects of ship configurations on survivability from an energy availability standpoint. Lastly, design tradeoffs among competing objective functions are examined.

6.1 ALL-ELECTRIC SHIP THERMAL AND ELECTRICAL LOADS

In the total-ship energy system model that is described in Chapter 5, user-defined load profiles are required as inputs to simulate time-dependent electrical, chilled water (CW), and refrigerated air (RA) demand. For the fuel consumption and survivability studies, these loads must represent the main operating modes that a ship undergoes, and they must include large thermal and electrical loads that are associated with the future electric ship components. Furthermore, the loads must be zone-specific, *i.e.* for each operating mode, the values of the loads in each of the five ship zones must be known. To achieve this objective, four types of loads are estimated and summed to form total-ship load scenarios: baseline service loads, propulsion loads, railgun loads, and phased array radar loads.

Detailed thermal and electrical load data for Navy ships either does not exist or is not publicly available. However, a limited data set was made available to the AES research community, and it is used to estimate the basic service loads of the ship. Basic

service loads include those associated with basic shipboard equipment and necessities such as lighting, computers, and kitchen appliances. In this data set, six notional operating modes that are typical of a Navy ship are present: Shore, Anchor, Low Speed, Cruise Speed, Sustained Battle, and Full Speed. The zonal electrical, CW, and RA loads for each operating mode are shown in Figures 6.1-6.3.

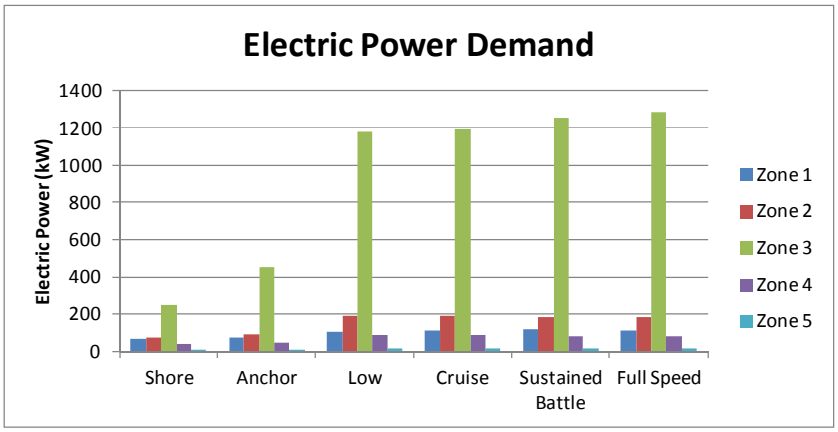


Figure 6.1: Electric power basic service loads

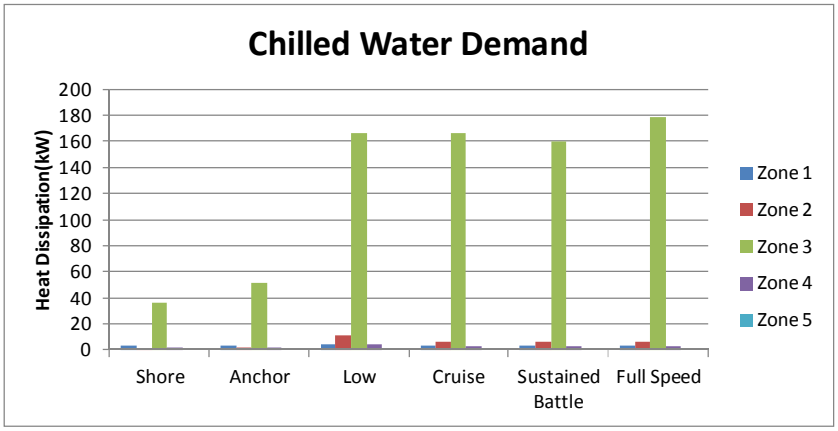


Figure 6.2: Chilled water basic service loads

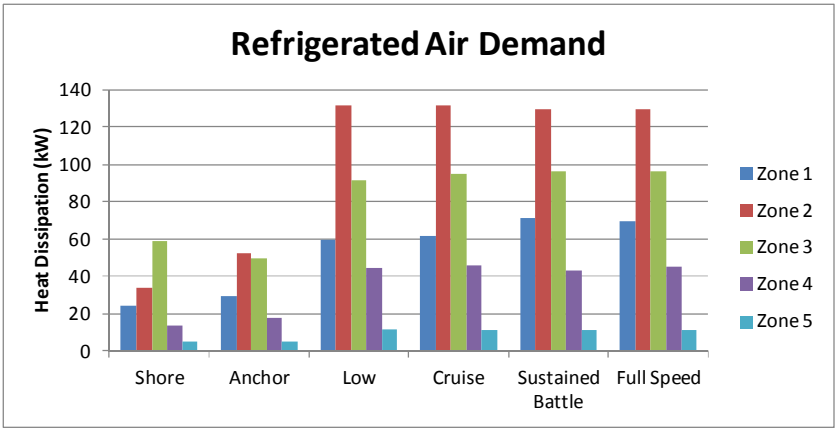


Figure 6.3: Refrigerated air basic service loads

The propellers on the AES are driven with induction motors that required large amounts of electric power to operate. In this research, propulsion loads are estimated using the power demand versus ship speed curve shown in Figure 6.4. Figure 6.4 shows that propulsion power is roughly proportional to the cube of the ship speed. Furthermore, propulsion power is very intense at full speed relative to the total installed electric power generation capacity of the gas turbine generators (81 MW).

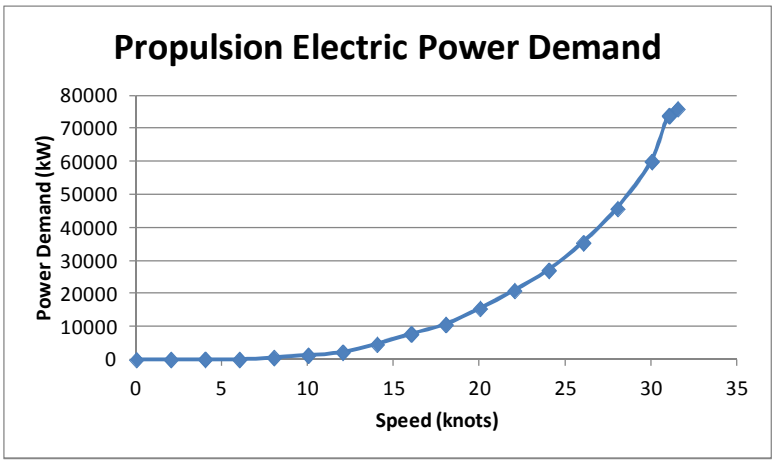


Figure 6.4: Propulsion electric power demand versus ship speed [13]

A constant ship speed is assumed for each of the six operating modes outlined above. These speeds and their corresponding electric power demand are listed in Table 6.1. The propulsion load is always assumed to originate in Zone 3 of the ship.

Table 6.1: Assumed speeds and propulsion power demands by operating mode

Mode	*Speed (knots)*	*Power Demand (kW)*
Shore	0	0
Anchor	0	0
Low Speed	16	7,800
Cruising	20	15,488
Sustained Battle	18	10,666
Full Speed	31	73,888

The next specialized load is a railgun. A railgun uses an electromagnetic force to fire its projectiles. Unlike conventional large-bore ammunition, a railgun projectile is an inert kinetic energy round. Therefore, there is no need to store large amounts of explosive propellant on the ship, and each round is significantly less expensive than conventional ammunition [111]. However, this process is highly power intensive, and it produces large amounts of waste heat that must be removed with chilled water. A single shot for a railgun is a highly dynamic event in which the peak power requirement is significantly larger than the 81 MW of installed gas turbine generation capacity on the AES. Therefore, specialized pulsed power equipment is assumed to be present on the ship that is capable of storing and rapidly discharging large amounts of electrical energy. In this research, average electrical and thermal requirements for continuous firing are estimated. A notional Naval railgun has an approximate electrical energy input to kinetic energy output efficiency of 40% and a muzzle energy of 64 MJ [112]. Therefore, the electrical energy required for a single shot is given by Eq. (6.1):

$$E_{electrical} = \frac{E_{muzzle}}{\eta_{rg}} = \frac{64}{0.4} = 160MJ \tag{6.1}$$

where $E_{electrical}$ is the electrical energy input, E_{muzzle} is the muzzle energy of the projectile, and η_{rg} is the overall efficiency of the railgun. Railguns have significant thermal loads that must be effectively managed by the CW system. The assumed efficiency of the railgun is used to estimate these loads using Eq. (6.2):

$$E_{loss} = \left(1 - \eta_{rg}\right) E_{electrical} = 96MJ \tag{6.2}$$

where E_{loss} is the total energy lost to railgun inefficiency. It is assumed that 25% of the energy lost is directly ejected from the bore of the weapon, and 75% of the energy lost is absorbed by the barrel and must be cooled with CW. Therefore, the thermal energy load for the weapon is given by Eq. (6.3):

$$E_{thermal} = 0.75 E_{loss} = 72MJ \tag{6.3}$$

where $E_{thermal}$ is the energy that must be removed with CW for each shot. To estimate the electric and CW power requirements, a firing rate of six rounds per minute is assumed. Therefore, the electric power and heat removal power requirements during continuous firing of the railgun are given by Eq. (6.4):

$$\begin{aligned} P_{electrical} &= 160\,{}^{MJ}\!/_{rnd.} \frac{6}{60}\,{}^{rnd}\!/_{s} = 16MW \\ P_{thermal} &= 72\,{}^{MJ}\!/_{rnd.} \frac{6}{60}\,{}^{rnd}\!/_{s} = 7.2MW \end{aligned} \tag{6.4}$$

The railgun loads are assumed to reside in Zone 1 and are only present during the Sustained Battle operating mode.

The final specialized load that is estimated for inclusion in the total-ship load scenarios is an active phased array radar. Phased array radar systems are composed of an arrangement of hundreds of small heat-producing electronic devices. The radar system is cooled with RA, where cold air from the RA system is forced through small holes in the face of the array. These holes form small jets that impinge directly on the individual

electrical components [113]. Warm air is ejected through the face of the array and circulated back through the RA system cooling coils. Estimates of thermal heat removal requirements of a military grade phased array are not publicly available. However, Price estimates that waste heat must be dissipated from a full system at a rate of "100 to many kW" [114]. For the purposes of this research, the thermal load of the phased array radar system is assumed to be 1,000 kW with system electrical conversion efficiency of 50%. Therefore, the electrical load of the phased array radar system is 2,000 kW. The radar system loads are assumed to reside in the superstructure of the ship (Zone 5), and the radar system is assumed to be running in all of the at-sea operating modes, *i.e.* all modes except Shore.

In Figures 6.5-6.7, the total estimated electric power, CW, and RA demands are shown by zone. These loads include the basic service loads, propulsion loads, railgun loads, and phased array radar system loads. As expected, propulsion dominates the electric power demand in the high speed operating modes. This behavior is seen in Figure 6.5, where Zone 3 (the zone in which the electric propulsion motors are housed) has very large loads during Full Speed operation. Furthermore, the largest CW demand is that of the rail gun, which is present as a large CW load in Zone 1 during the Sustained Battle mode. Lastly, RA demands are dominated by the radar system, which are present in Zone 5 in all at-sea modes.

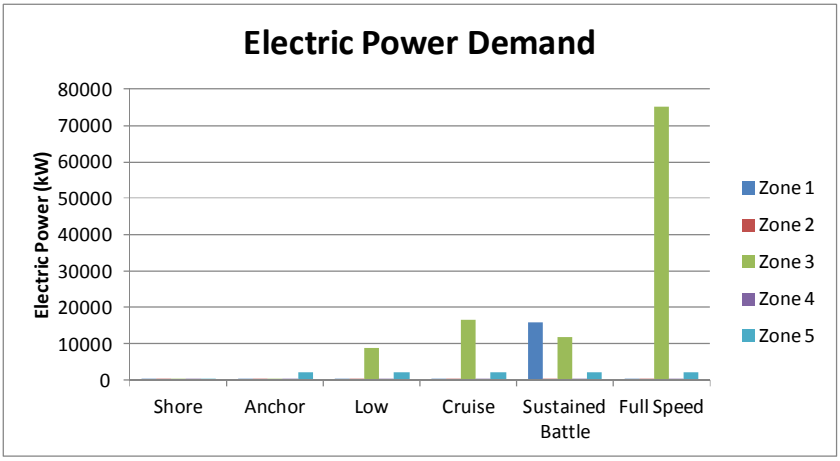


Figure 6.5: Zonal electric power demand by operating mode

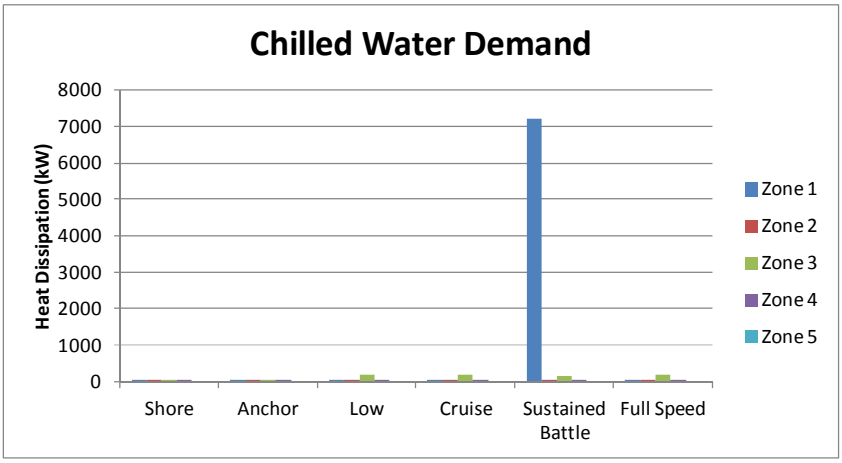


Figure 6.6: Zonal chilled water demand by operating mode

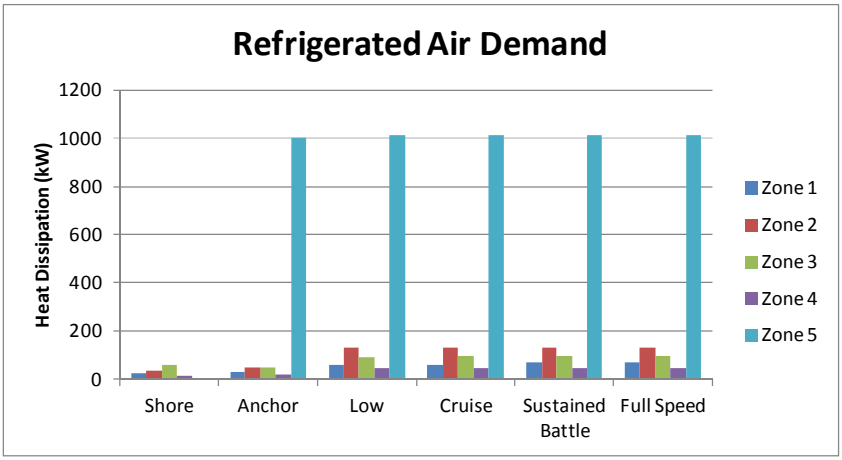


Figure 6.7: Zonal refrigerated air demand by operating mode

6.2 FUEL CONSUMPTION STUDY

6.2.1 Motivation

In this section, AES energy system configurations are sought that aim to reduce average annual fuel consumption. Fuel has a major impact on the cost of operating a Navy ship throughout its lifecycle. For example, a one percent reduction in fuel consumption in DDG 51 class ships results in an estimated annual fuel cost saving of approximately $100k per ship [115].

The configuration of the energy system components and the topologies of the electric power and CW ZSDs affect the rate of fuel consumption. For example, the connectivity of the CW system affects the friction losses in the CW distribution pipes because inter-zone piping must be used to transport CW greater distances across zone boundaries when more zones are connected. Therefore, it is advantageous to produce CW locally to meet in-zone demand whenever possible. The electrical ZSDs also affect fuel consumption due to their impact on generator operating levels. The total-ship energy system model described in Chapter 5 automatically runs all available generators in a connected set of zones to be as efficient as possible. That is, a generator is not online unless it is needed to meet demand. Otherwise, it is turned off and the other generators meet demand while operating at a higher percent of their rated load. Standard operating practice dictates that all online connected generators must operate at the same percentage of their rated load. The MT30 (36 MW) generators have a lower specific fuel consumption than the RR4500 (4.5 MW) generators, on average. Furthermore, the MT30s have significantly higher capacity than the RR4500s, and operating them at or near their most efficient percent of rated load (100%) has a more weighted effect on fuel efficiency than doing so with the RR4500s. Therefore, segregating a zone or set of zones

that contain the MT30s may result in a reduced rate of fuel consumption if doing so enables the larger generators to operate closer to 100% of their rated load.

Identifying the configuration that results in the lowest fuel consumption is not an easy task. First, the number of configurations to choose from is large, warranting the use of automated search techniques such as the CGS method. Second, the AES load scenarios are diverse, and the best configuration for one operating mode may not be the best configuration for all others. Lastly, equipment space below deck is extremely limited, and the best configuration may not even be feasible due to strict volumetric size constraints. Therefore, the goal of this study is to use the CGS method to identify configurations and corresponding electrical and CW ZSD topologies that result in reduced average fuel consumption over the lifecycle of the ship.

6.2.2 Problem Setup

In this section, the loads, variables, constraints, and the objective function evaluation of the fuel consumption study are explained. The total-ship energy model that is used to assess fuel consumption requires user-defined electrical, CW, and RA loads. The load profile used for the fuel consumption study used here features all six of the AES operating modes. Steady-state fuel consumption is evaluated for each of the six operating modes, and each operating mode is weighted according to an estimation of the annual time spent in each mode, shown in Table 6.2.

Table 6.2: Percent of time spent in each mode annually

	Shore	*Anchor*	*Low Speed*	*Cruise*	*Sustained Battle*	*Full Speed*
Percent	15%	35%	18%	27%	2%	3%

The configuration variables in the fuel consumption study include the quantity and locations of gas turbine generators and CW plants. There are exactly two MT30s,

two RR4500s, and thirteen CW plants available to be placed throughout the ship. Thirteen CW plants are allowed because that is the exact number required to meet thermal loads in the most extreme load scenario. Storage is not considered in the fuel consumption study. Storage provides temporary additional power when rapid load changes occur and when damage exists on the ship. However, for the fuel consumption study, the loads are assumed to be steady state and all components and distribution buses are undamaged and fully operational.

The number of possible configurations can be significantly reduced *a priori* by referring to the volumetric space constraints that are presented in Appendix B. In particular, MT30s are quite large and can only be placed in Zones 2 and 3 due to volumetric constraints. RR4500s can only be placed in Zones 1-4, and only one RR4500 will fit in Zone 1. Lastly, CW plants can fit in all five zones. However, only two CW plants can fit in the superstructure (Zone 5) of the ship. By adhering to these constraints, a summary of the variables and the range of discrete values they can assume is given in Table 6.3.

Table 6.3: Fuel consumption study configuration variables

x_n	*Description*	*Possible Values*	*Notes*
x_1	MT30s in Z2	{0,1,2}	MT30s only fit in Z2 and Z3
x_2	MT30s in Z3	{0,1,2}	MT30s only fit in Z2 and Z3
x_3	RR4500s in Z1	{0,1}	Only 1 RR4500 fits in Z1
x_4	RR4500s in Z2	{0,1,2}	
x_5	RR4500s in Z3	{0,1,2}	
x_6	RR4500s in Z4	{0,1,2}	
x_7	CW Plants in Z1	{0,1,2,…,13}	
x_8	CW Plants in Z2	{0,1,2,…,13}	
x_9	CW Plants in Z3	{0,1,2,…,13}	
x_{10}	CW Plants in Z4	{0,1,2,…,13}	
x_{11}	CW Plants in Z5	{0,1,2}	Only 2 CW Plants fit in Z5

Although the variables in this problem can be equal to any of the values in the "Possible Values" column in Table 6.3, there is an additional constraint that must be satisfied. The total number of each type of component must be exactly equal to predetermined values (two MT30s, two RR4500s, and thirteen CW plants). For example, x_1 cannot be equal to 2 of x_2 is equal to 1, because such a solution would imply that there are three MT30s on the ship. This problem is similar to the traveling salesperson problem that is solved with the CGS method in Chapter 4; it is a combinatorial optimization problem with discrete variables and a discontinuous response. There are exactly 37,233 possible solutions to this problem, not accounting for unforeseen volumetric constraint violations.

The ZSD topology is not a design variable to be optimized because the ZSD positions can be switched on and off after the ship is in service, while the configuration of the ship is permanent and cannot be changed once the ship has been constructed. However, the ability to change the connectivity of the ship with the ZSDs has significant implications on the most fuel efficient component configuration. For each evaluation of the objective function, all combinations of the CW and electric power ZSDs are searched

to find the topology that results in the least fuel consumption without allowing any unmet demand. This task is performed separately for each of the six operating modes. The CW ZSD topology that results in the least amount of electric power demand is determined first. The process is repeated using the CW ZSD topologies that minimize the electric power demand of the CW system. In this case, electrical ZSD topologies are identified that result in the least amount of fuel consumption without allowing any unmet demand. This process is performed first for the CW system because electric power demand depends on the CW system configuration and topology. Finally, a single simulation is run for each operating mode using the predetermined best CW and electric power ZSD topologies for each operating mode. The total fuel consumption is recorded, with the time spent in each operating mode weighted according to the percentages listed in Table 6.2.

6.2.3 Fuel Consumption Study Results and Discussion

Rate of convergence tests are performed for the CGS method, genetic algorithms (GAs), and random search in a manner similar to those for the three discrete variable test problems presented in Chapter 4. Due to the computational expense of a single objective function evaluation (approximately 4 minutes), the three methods are executed five times each with the maximum number of objective function evaluations set at 500. The average best known solution versus the number of function evaluations for the three methods is presented in Figure 6.8. The upper and lower error bars at select points indicate the maximum and minimum objective function values found by the five tests with their respective numbers of function evaluations.

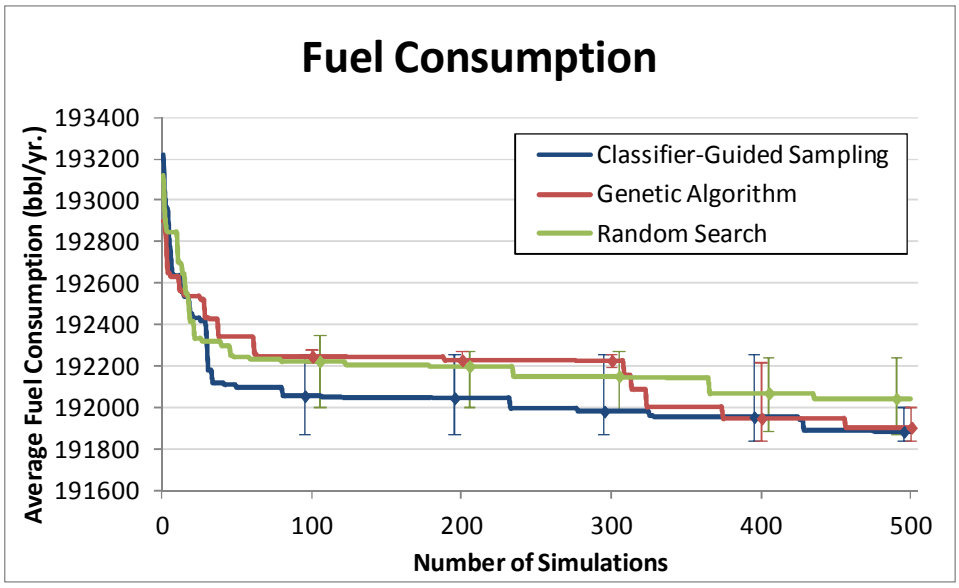


Figure 6.8: Fuel consumption study results

On average, the CGS method finds significantly better solutions than the random search method, and slightly better solutions than the GA. The error bars indicate that the best solutions found with the CGS method and the GA are only slightly better than the best solution found with random search. However, the worst solution found with the CGS method is significantly better than the worst solutions found with random search. For the first 300 simulations, the GA struggles to make improvements beyond the randomly generated initial population. However, convergence improves significantly after 300 simulations are performed, and the average best solution found with the GA is only slightly worse than that found with the CGS method.

The best solutions found with the CGS method may appear to be only marginally better than those found with random search. However, the metric for the objective function is the average fuel consumption of the ship in barrels per year, and it includes all of the modes of operation that the ship is expected to undergo over its lifecycle. Assuming $175/bbl [115], the difference between the average of the best solutions found

with random search and the average of the best solutions found with the CGS algorithm amounts to approximately $29,200 per year.

The best overall solution is found with two of the five tests using the CGS method. This solution, its resulting average rate of fuel consumption, and ZSD positions are shown in Table 6.4. A schematic of the configuration implied by the solution in Table 6.4 is shown in Figure 6.9.

The best known solution places the two MT30 generators in Zone 3 and one each of the RR4500 generators in Zones 1 and 4. Furthermore, there are two CW plants in Zone 1, one CW plant in Zone 3, eight CW plants in Zone 4, and two CW plants in Zone 5. Interestingly, there are no components placed in Zone 2. However, placing an RR4500 in Zone 2 instead of Zone 1 yields the same result.

Table 6.4: Best known fuel consumption solution

Category	*Component / Mode*	*Z1*	*Z2*	*Z3*	*Z4*	*Z5*
Configuration Variables	RR4500s	1	0	0	1	0
	MT30s	0	0	2	0	0
	CW Plants	2	0	1	8	2
Electrical ZSD Topology	Shore	1	1	1	1	1
	Anchor	1	1	1	1	1
	Low Speed	1	1	1	1	1
	Cruise	1	1	1	1	1
	Sustained Battle	1	1	1	1	1
	Full Speed	1	0	1	1	0
Chilled Water ZSD Topology	Shore	1	1	0	1	1
	Anchor	1	1	0	1	1
	Low Speed	1	0	1	0	1
	Cruise	1	0	1	0	1
	Sustained Battle	1	1	1	1	1
	Full Speed	1	0	1	0	1
Objective Function Value						
Average rate of fuel consumption (bbl/yr.):				191,843.6		

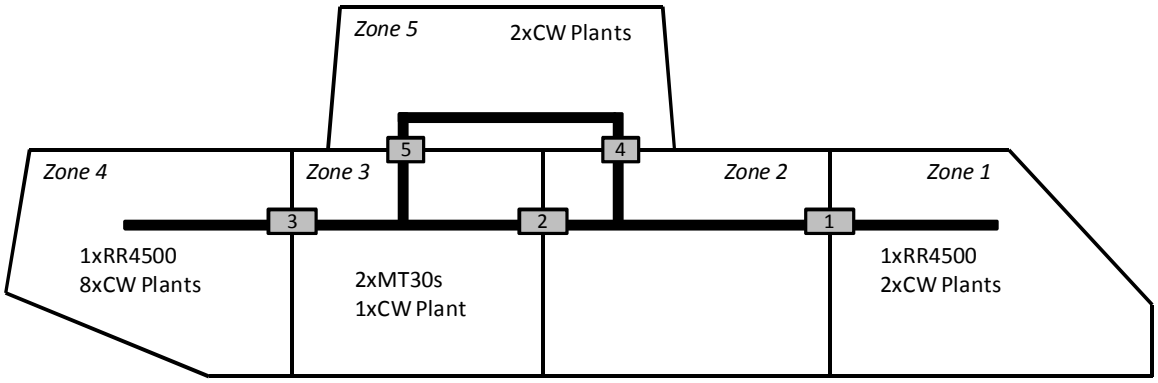


Figure 6.9: Schematic of best configuration for fuel consumption

In five of the six operating modes, all of the electrical system ZSDs are in the ON position because this topology is the only one that results in zero unmet demand. However, in the full speed mode, ZSDs 2 and 5 are in the OFF position. This topology results in two sets of connected zones: Zones 3 and 4 are one connected set of zones and Zones 1, 2 and 5 are the other connected set. There are approximately 77.6 MW of electric power demand in the Full Speed operating mode (not including the electric power required for CW and RA generation). Of this total, 75.3 MW of the demand is in Zones 3 and 4, and a vast majority of this demand is in Zone 3 in the form of propulsion. It is best to operate the large generators (the MT30s) as efficiently as possible even if it means operating one or two of the smaller generators at a less efficient operating percent. In this case, segregating Zones 3 and 4 enables the two MT30s and one of the RR4500s to operate at 98.4% of their rated load. However, if the ship were fully connected (all ZSDs on), all of the ship generators would be running at 95.8% of their rated load.

A somewhat counterintuitive feature of the best known configuration is that there are eight CW plants in Zone 4, which is the most aft section of the ship. However, the largest thermal loads are those that are associated with the railgun and they are in Zone 1. This type of configuration is not consistent among the five best unique solutions found

with the CGS method (Appendix C). In other configurations that perform similarly, there are 5, 6, or 8 CW plants placed in Zone 1. This result is due to the fact that the railgun firing mode makes up a very small percentage of total operating time (2%), and the most efficient configuration for this mode is lightly weighted. Therefore, the railgun thermal loads have a relatively insignificant effect on overall fuel consumption.

The five best unique solutions found with the CGS algorithm are presented in Appendix C. There are several similarities among the five solutions. First, all five of the configurations place two MT30s in Zone 3 and one RR4500 in Zone 4. This trend highlights the need for Zones 3 and 4 to produce 76.5 MW of power generation capacity (two MT30s plus one RR4500) as a segregated set of zones during full speed operation. Second, all five of the configurations place two CW water plants in Zone 5. The second highest thermal load on the ship is that of the radar system, which requires approximately 1,000 kW of heat removal capacity in that zone. This heat is removed with RA, and RA is produced through cooling coils with CW. Each CW plant produces approximately 700 kW of heat removal capacity, and placing two CW plants in Zone 5 enables local production of all of the required heat removal capacity for the radar system.

6.3 SURVIVABILITY STUDY

6.3.1 Motivation

Survivability is a term than can have many meanings in terms of warship design and operation. In this research, survivability is defined as the ability of a shipboard energy system "to support the ship's ability to continue fulfilling its mission to the degree planned for the particular threat…" [10]. The ability to continue to fulfill a particular mission depends on many factors, including the physical integrity of the ship structure, availability of personnel, and functionality of technological systems. In this research,

survivability is assessed from the standpoint of energy availability. Survivability is evaluated by subjecting the ship to a set of notional damage scenarios in which energy production components and their respective distribution architectures are compromised. Survivability is measured by recording the amount of unmet electric power demand or thermal heat dissipation demand during notional damage scenarios.

The locations of energy production and storage components have a significant effect on survivability. For example, the railgun is located in Zone 1 of the ship and has significant electrical and thermal loads associated with it. The railgun is also used when the ship is in the Sustained Battle mode, which is a mode during which the ship is at a heightened risk of sustaining damage. If the distribution architecture were damaged and electric power could not be transported to Zone 1 from, say, Zone 3, the railgun would not be able to perform its intended function. Therefore, in this example, the ship would be more survivable if electric power were produced locally.

6.3.2 Problem Setup

In this section, the loads, damage scenarios, variables, constraints, and objective function evaluations of the survivability study are explained. Unlike the fuel consumption study, in which all six of the operating modes are tested and weighted by percent of time spent in each mode, only the Sustained Battle and Full Speed modes are included in the load profile for the survivability tests. A notional real-life dynamic event is simulated, in which a ship is attacked and sustains damage. Immediately following the attack, the ship is operated in Full Speed mode for five minutes, followed by Sustained Battle operation for another five minutes. This dynamic load profile simulates an event in which the ship must move quickly to an advantageous counterattack position and retaliate.

This load scenario is tested for four distinct damage scenarios. In each damage scenario, one of the below-deck zones (Zones 1-4) is damaged. When damage occurs to a zone, half (rounded down) of the installed electric power generation, CW production, or energy storage components in that zone are immediately unavailable. Furthermore, any horizontal ZSDs (ZSDs 1, 2, or 3) that are adjacent to the damaged zone are immediately turned to the OFF position. Vertical ZSDs (ZSDs 4 and 5) are unaffected. During each damage scenario, all of the unaffected ZSDs are immediately opened to maximize connectivity of the ship. These four damage scenarios simulate the effects of an attack on energy production, storage, and distribution equipment throughout the length of the ship.

The configuration variables for the survivability study include the quantity and locations of gas turbine generators, CW plants, and flywheels for energy storage. Flywheels were selected as the means of energy storage for the survivability study due to their compact size and greater power density than can be offered by battery storage options. The flywheels are available to be placed in pairs in Zones 1, 4, and 5. One pair of flywheels offers 1,250 kW of charge / discharge rate and 104 kW-hr of storage capacity. These specs are consistent with a notional Navy ship flywheel configuration described by Hebner *et al.* [116].

The configuration variables for the survivability study are similar, but not identical, to those used in the fuel consumption study. To reduce the size of the design space, it is assumed that the two MT30 generators are placed in Zone 3 because all five CGS runs of the fuel consumption study suggest a configuration with this feature. The same volumetric constraints apply to the survivability study as are applied to the fuel consumption study (Appendix B). To further reduce the design space, no RR4500s can be placed in Zone 3 because there are already two MT30s in that zone, and placing an RR4500 there violates volumetric space constraints. Therefore, RR4500s can only be

placed in Zones 1, 2, or 4, because they are also too large for Zone 5. CW plants can be placed in all five of the ship zones, but there is only space for two CW plants or two flywheels in Zone 5. A summary of the variables and the ranges of discrete variables that they can assume is given in Table 6.5.

Table 6.5: Survivability study configuration variables

x_n	*Description*	*Possible Values*	*Notes*
x_1	RR4500s in Z1	{0,1}	Only 1 RR4500 fits in Z1
x_2	RR4500s in Z2	{0,1,2}	
x_3	RR4500s in Z4	{0,1,2}	
x_4	CW Plants in Z1	{0,1,2,…,13}	
x_5	CW Plants in Z2	{0,1,2,…,13}	
x_6	CW Plants in Z3	{0,1,2,…,13}	
x_7	CW Plants in Z4	{0,1,2,…,13}	
x_8	CW Plants in Z5	{0,1,2}	Only 2 CW Plants fit in Z5
x_9	Flywheels in Z1	{0,2,4,6,8}	
x_{10}	Flywheels in Z4	{0,2,4,6,8}	
x_{11}	Flywheels in Z5	{0,2}	Only 2 flywheels fit in Z5

There are exactly two RR4500s, two MT30s, and thirteen CW plants that are available to be placed throughout the ship. However, storage components are considered optional, because they are not necessary to meet steady state undamaged demand. Therefore, there is no set requirement on the number of flywheels that are included in the configuration, and there can be anywhere from zero to eight flywheels (in pairs) in their allowed zones of the ship. There are 172,375 possible solutions to this problem, not accounting for unforeseen volumetric constraints.

A single evaluation of the objective function simulates all four of the damage scenarios described above. This task is achieved by running the ten minute simulation once for each damage scenario. For each simulation, the total unmet demand of all zones, including electrical and thermal dissipation, is recorded. This information is used to assess survivability in two ways. In the first way, termed “Extreme Survivability”, the

maximum unmet demand of all four damage scenarios is given as the objective function output, as shown in Eq. (6.5):

$$f_{extreme} = \max\left(\left[u_1, u_2, u_3, u_4\right]\right) \tag{6.5}$$

where u_i is the total unmet demand for damage scenario i. The second survivability metric, termed "Robust Survivability", is the average plus the standard deviation of the unmet demand of all four damage scenarios, as shown in Eq. (6.6):

$$f_{robust} = \bar{u} + s_u \tag{6.6}$$

where $\bar{u}$ and s_u are the mean and standard deviation of the total unmet demands for all damage scenarios.

6.3.3 Extreme Survivability Results and Discussion

The CGS method, a GA, and random search are used to perform rate of convergence tests for the two survivability objective functions described above. In Figure 6.10, the results of the extreme survivability rate of convergence test using the three methods are presented. The CGS method, GA, and random search tests are performed five times each and the average best known solution versus the number of objective function evaluations is presented. The upper and lower error bars at select points indicate the maximum and minimum objective function values found by the five tests with their respective numbers of function evaluations.

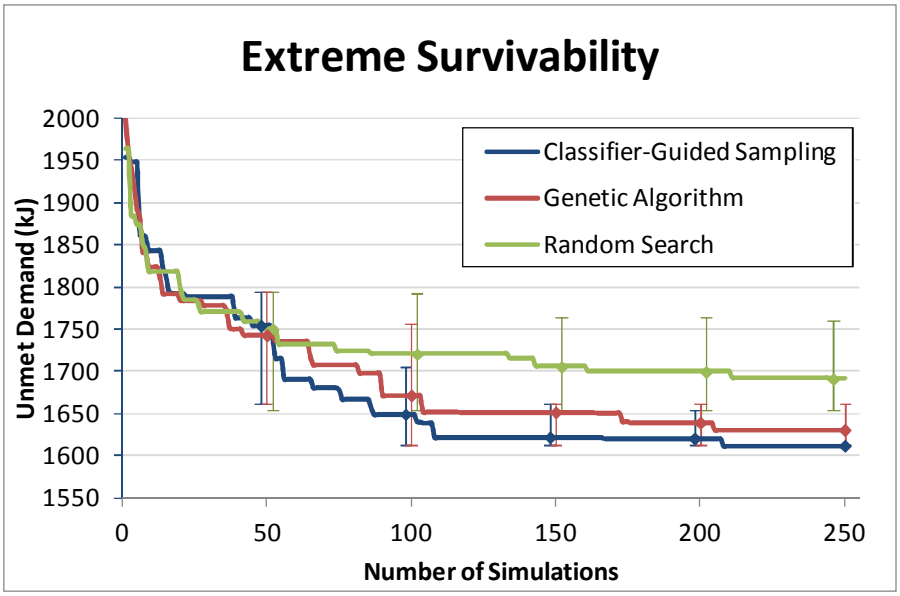


Figure 6.10: Extreme survivability study results

On average, the CGS method finds better solutions than the random search method and the GA. Furthermore, the error bars indicate that the best solution found with random search is worse than the worst solution found with the CGS method. In fact, there are no error bars by the time the maximum number of function evaluations is reached for the CGS method because the CGS method found solutions that yield identical objective function values with only 250 evaluations in all five trials. However, the GA only converges to similar solutions in three of the five trials. Inspection of the five best unique solutions found (Appendix C) indicates that several solutions yield the same objective function value in this case. One of these five solutions is presented in Table 6.6, along with the unmet demand by zone and damage scenario for the electrical and heat dissipation systems. A schematic of the solution implied by Table 6.6 is shown in Figure 6.11.

Table 6.6: Extreme survivability objective solution

Category	*Component / Damage*	*Z1*	*Z2*	*Z3*	*Z4*	*Z5*
Configuration Variables	RR4500s	0	2	0	0	0
	MT30s	0	0	2	0	0
	CW Plants	10	1	0	1	1
	Flywheels	8	0	0	0	0
Electrical Unmet Demand (kJ)	Damage Scenario 1	1,207	0.3	89.9	0.1	2.5
	Damage Scenario 2	1,062	0.2	95.6	0.2	2.7
	Damage Scenario 3	0.6	0.4	148.1	14.9	4.0
	Damage Scenario 4	0.3	0.2	77.9	12.8	2.1
Heat Dissipation Unmet Demand (kJ)	Damage Scenario 1	312.5	0	0	0	0
	Damage Scenario 2	20.2	0.9	1.8	0.3	6.9
	Damage Scenario 3	17.1	0.3	0.6	0	2.4
	Damage Scenario 4	17.1	0.3	0.6	7.9	2.4
Objective Function Value						
Maximum Total Unmet Demand (kJ):				1,612.1		

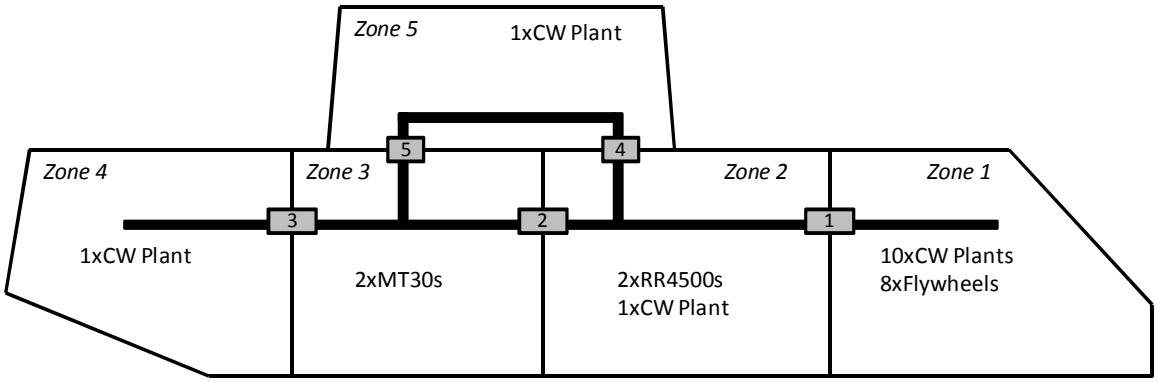


Figure 6.11: Schematic of best configuration for extreme survivability

In the configuration shown above, there are two features which are particularly noteworthy. All of the flywheels are placed in Zone 1, and ten of the thirteen CW plants are also placed in Zone 1. Zone 1, where the thermally and electrically intense railgun is located, is at the far end of the ship and is particularly susceptible to unwanted, damage-induced isolation. When damage occurs in Zones 1 or 2, ZSD 1 is set to the OFF position to simulate the local damage to the distribution network. Therefore, maximum unmet

demand is effectively reduced by placing large amounts of storage and CW production capacity in Zone 1. In all five of the CGS test runs, the best solution identified with the CGS method had these same features. The vulnerability of the railguns clearly dominates the maximum unmet demand, and the algorithm targets this issue in all test cases.

6.3.4 Robust Survivability Results and Discussion

In Figure 6.12, the rate of convergence test results for the robust survivability study are presented. The upper and lower error bars indicate the best and worst solutions that the CGS, a GA, and random search method are able to find among the five trials.

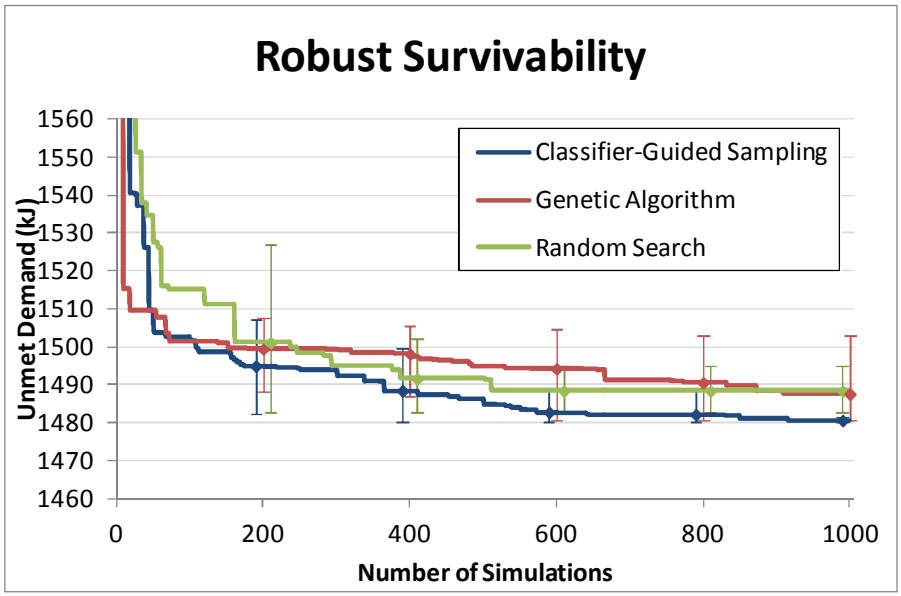


Figure 6.12: Robust survivability study results

For the robust survivability study, the CGS method performs better, on average, than the random search method and the GA. However, the CGS method performs only marginally better than the competing methods. Between approximately 250 and 900 simulations, the GA performs worse than the Random search method, and its average performance is only slightly better than that of random search at the very end of the trials.

Inspection of the GA error bars at 1,000 simulations indicates that the GA performed very poorly in one of the five trials, thus raising the average objective function value significantly. The error bars indicate that the worst solution the CGS method finds is worse than the average of the solutions found by random search when the number of simulations is less than or equal to 800. By the time 1,000 simulations are performed, the CGS method identifies five similar solutions that are all superior to the best solution that random search finds.

The best solution that the CGS method identifies for the robust survivability problem is given in Table 6.7, along with the unmet demands for each zone and damage scenario for the electric power and thermal systems. A schematic of this configuration is shown in Figure 6.12.

Table 6.7: Best known robust survivability objective solution

Category	*Component / Damage*	*Z1*	*Z2*	*Z3*	*Z4*	*Z5*
Configuration Variables	RR4500s	1	1	0	0	0
	MT30s	0	0	2	0	0
	CW Plants	7	6	0	0	0
	Flywheels	0	0	0	6	2
Electrical Unmet Demand (kJ)	Damage Scenario 1	1,376	0.4	83.2	0.1	2.2
	Damage Scenario 2	85.2	0.4	88.9	0.1	2.4
	Damage Scenario 3	0.5	0.5	164.2	0	4.4
	Damage Scenario 4	0.2	0.3	86.9	0	2.3
Heat Dissipation Unmet Demand (kJ)	Damage Scenario 1	429.7	0	0	0	0
	Damage Scenario 2	195.5	0	0	0	0
	Damage Scenario 3	0	0	0	7.9	0
	Damage Scenario 4	0	0	0	7.9	0
Objective Function Value						
Mean Plus Std. Dev. of Total Unmet Demand (kJ):				1,480.3		

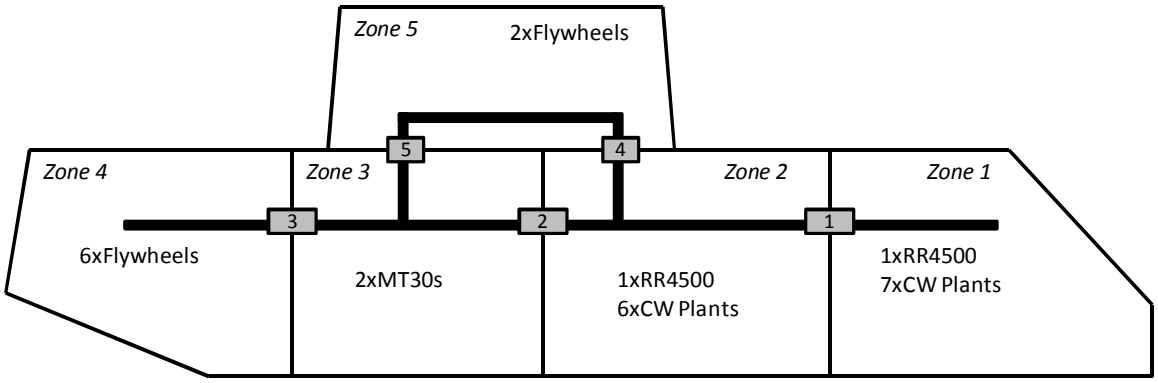


Figure 6.12: Schematic of best configuration for robust survivability

The best configuration identified for the robust survivability objective function has some very significant differences from that which is found for the extreme survivability objective function. In this configuration, there are no flywheels in Zone 1. Rather, there are six flywheels in Zone 4 and two flywheels in Zone 5. All of the CW plants are placed near the front of the ship, with seven of them placed in Zone 1 and six of them placed in Zone 2.

For this configuration, unmet demand for damage scenario 1 (Zone 1 damaged) is quite large, especially in Zone 1 where there is no storage and one RR4500. During damage scenario 1, the RR4500 in Zone 1 is damaged and offline, and only 3 of the CW plants are available. However, there are large thermal and electrical loads associated with the railgun in this damage scenario. This is the only damage scenario with excessive unmet demand, and unmet demand for the other damage scenarios is significantly less than it is for damage scenario 1. When compared to the results for the extreme survivability study, thermal unmet demand is now significantly higher in Zone 1 for both damage scenarios 1 and 2. However, the unmet demand is either reduced or eliminated in nearly all of the other zones and damage scenarios. The placement of flywheels in Zones 4 and 5 helps to serve the large electric power demand in Zone 3 that is associated

with propulsion during Full Speed mode. In particular, during damage scenario 2 when Zone 2 is damaged, none of the RR4500s are available to serve this zone because they are either damaged or isolated due to damaged distribution busses. However, the electrical loads in Zone 3 for this operating mode cannot be met with two MT30s alone, and the flywheels in Zones 4 and 5 help serve this load when Zone 2 is damaged.

There is never any unmet thermal demand in Zone 5 even though there are no CW plants in that zone. This is because the worst case, as far as CW availability to this zone is concerned, is damage scenario 2. In this scenario, Zones 2, 3, 4, and 5 must share only 3 CW plants (ZSDs 4 and 5 are undamaged in this scenario). However, thermal loads in Zones 2-4 are relatively small, and three CW plants provide enough heat dissipation capacity to serve the thermal loads in Zones 2-5 even with the large thermal loads in Zone 5 from the radar system.

6.4 MULTIOBJECTIVE CONSIDERATIONS AND DESIGN TRADEOFFS

The fuel consumption and survivability studies presented in the previous sections produce results that are drastically different in comparison to each other. These objective functions contradict each other, and tradeoffs exist that must be considered when deciding between competing AES energy system configurations. In Figures 6.13 and 6.14, the best five unique configurations for each objective function are plotted.

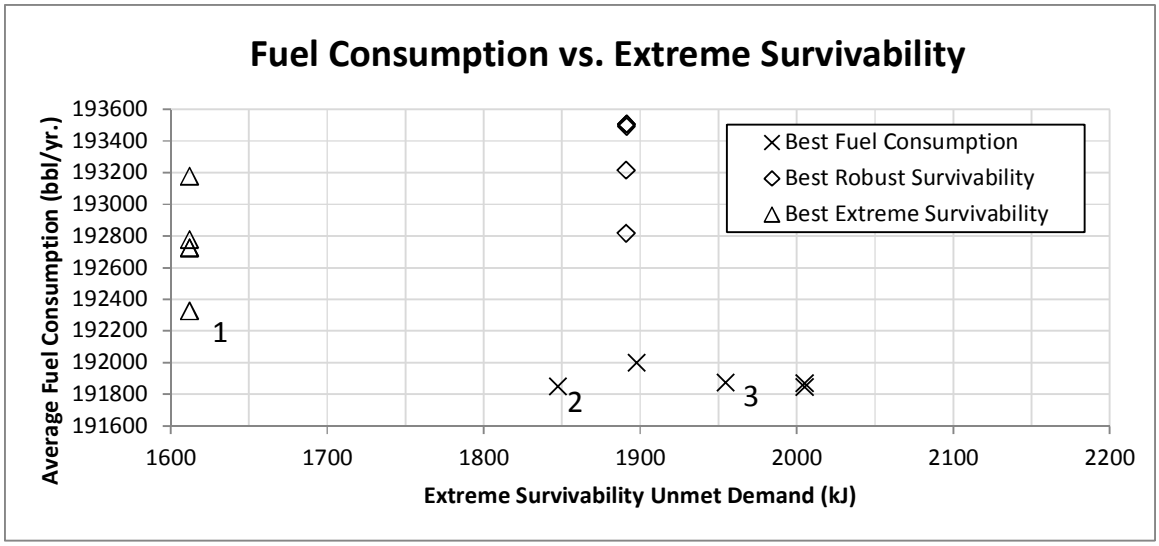


Figure 6.13: Fuel consumption versus extreme survivability

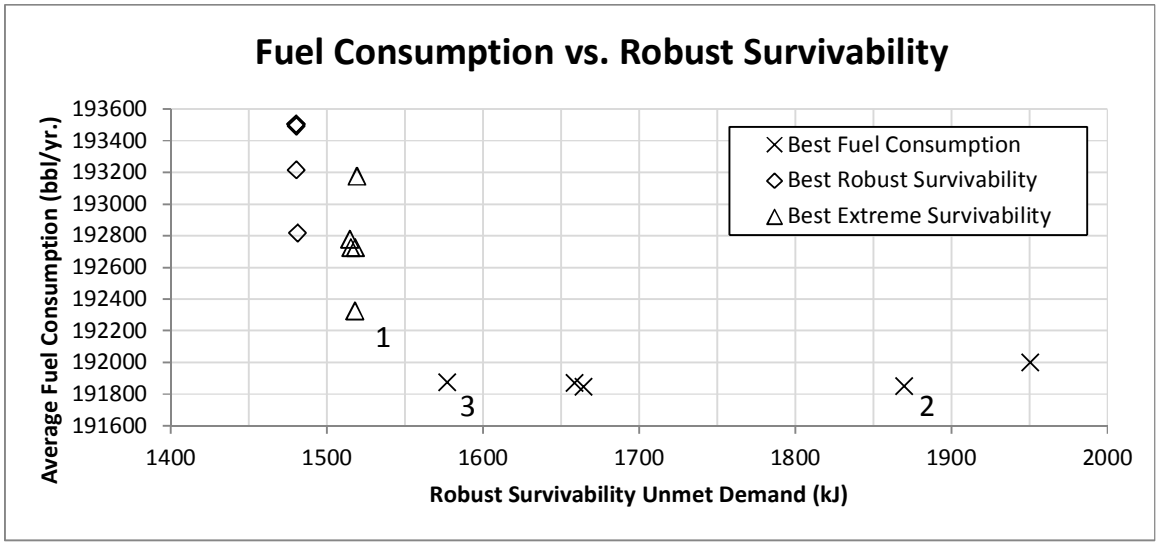


Figure 6.14: Fuel consumption versus extreme survivability

In Figure 6.13, the average fuel consumption performance versus the extreme survivability unmet demand (maximum of the four damage scenarios) is plotted. Based on the figure, there are two configurations that perform well for the two objective functions considered. The first configuration, labeled as configuration "1", is the configuration that is optimized for extreme survivability with the least fuel consumption.

All of the other extreme survivability configurations have higher rates of fuel consumption. The second promising configuration, labeled as configuration "2", is the configuration that is optimized for fuel consumption and has the least extreme survivability unmet demand. There are two configurations that have slightly better fuel consumption performance, but the difference is so slight that the benefits of reduced unmet demand for extreme survivability outweigh the cost of slightly increased fuel consumption. Any of the configurations that are optimized for robust survivability perform worse than these two configurations when evaluated with the fuel consumption and extreme survivability objective functions.

In Figure 6.14, the average fuel consumption performance versus the robust survivability unmet demand (mean plus standard deviation of the four damage scenarios) is plotted. The configurations labeled "1" and "2" in Figure 6.13 are relabeled again in this figure. A third promising configuration, labeled as configuration "3" in both figures, is identified. Configuration "3" is the configuration that is optimized for fuel consumption configurations that performs best when evaluated with the robust survivability objective function. Although it performs relatively well on the robust survivability objective, it performs somewhat poorly when evaluated with the extreme survivability objective, referring back to Figure 6.13. The configurations that are optimized for extreme survivability perform only slightly worse than those that are optimized for robust survivability on the robust survivability metric. Therefore, designing for extreme survivability is shown to be preferred because designing to minimize average unmet demand to a variety of damage scenarios results in configurations that perform very poorly when evaluated with the extreme survivability and fuel consumption objective functions.

In conclusion, configuration "1" is likely the best candidate of all fifteen configurations plotted in Figures 6.13 and 6.14. It has the sixth best rate of fuel consumption and is only inferior, fuel consumption wise, to the configurations that are specifically optimized with the goal of reducing fuel consumption. This configuration also results in the least amount of maximum unmet demand of all four damage scenarios. Lastly, it performs only slightly worse in the robust survivability metric when compared to the best robust survivability configurations. This configuration is tabulated in Table C.9 in appendix C.

6.5 Chapter Summary and Design Recommendations

In this chapter, the CGS method is used on the total-ship energy system model to identify ship configurations that are more survivable and consume less fuel. Fuel consumption is assessed by subjecting the model to six distinct load scenarios that represent the full spectrum of operating modes that a Navy ship undergoes. Fuel consumption is subsequently calculated as the average rate of fuel consumption in barrels per year. To assess survivability, a load profile that simulates full speed operation immediately followed by counterattack is used. Generation and storage components are forced to become unavailable in the model to simulate damage. Survivability is assessed using two difference metrics. The first survivability metric measures the maximum amount of unmet demand across four different damage scenarios. The second survivability metric uses the average unmet demand skewed using the standard deviation to penalize configurations that have large variance among performance across damage scenarios. Several configurations are identified that perform well for each of the objective functions. However, an investigation of the tradeoffs among competing configurations indicates that it is possible to optimize the energy system configuration for

extreme survivability without significantly reducing fuel consumption and robust survivability performance.

Chapter 7: Conclusions and Future Work

Significant challenges are associated with early-stage design of next-generation naval warships. Engineers who are tasked with developing conceptual layouts of new ship designs face an overwhelming array of possibilities, and the ability to explore this design space rapidly and effectively is critical if engineers are to identify feasible, high-performance solutions in a reasonable amount of time. Computational expense and incompatibility of subsystem models hinders the process of exploring these vast design spaces. Furthermore, the discrete nature of categorical design variables and their discontinuous effects on performance metrics creates a need for specialized approximation and optimization tools.

The focus of this research is on total-ship thermal and electrical energy system design for the U.S. Navy's all-electric ship (AES). Determining which energy system components should be included in a ship, where they should be placed, and how they should be connected during varying operational scenarios is not an easy task, and the configuration of these components has significant implications for fuel consumption and survivability. This research aims to address the challenges associated with high-dimensional design spaces, discrete variables, discontinuous responses, and complex interactions between shipboard energy systems. In this chapter, a summary of this research is presented, along with a discussion of the contributions and suggestions for future work.

7.1 Summary of Research

The central hypothesis of this research is that continuous variable metamodels and the discrete variable classifier-guided sampling method (CGS) can be used as part of a system-level design framework to enable rapid design optimization of computationally

expensive shipboard energy distribution system models. This hypothesis is addressed by completing several tasks.

The first task performed in this research is to investigate metamodeling techniques as a means to provide fast approximations of computationally expensive component or subsystem models. Published literature suggests that the performance and scalability of metamodeling techniques is well-understood for approximating functions with low numbers of independent variables. However, relatively little is known about how these techniques perform when applied to functions with 15 or more variables. To fill this research gap, the prediction accuracies, training times, and prediction times of the kriging, support vector regression, and radial basis function metamodeling techniques are tested using a set of high-dimensional test problems. The results of this study suggest that the three methods have their own advantages and disadvantages, and this task provides practical insights about the applications for which each method is most appropriate. For example, kriging approximates all of the test functions in this study accurately with fewer or equivalent numbers of training points, relative to support vector regression (SVR) and radial basis functions (RBF). RBF metamodels are found to have extremely slow build times when the number of training points is large, while kriging has the slowest prediction speeds. However, SVR consistently has the fastest build and prediction times. Another advantage of kriging is its ease of use, because the correlation function parameters are automatically tuned during the training process. This is not the case with SVR and RBF, and the quality of the resulting fit is very sensitive to the user-defined tuning parameters of these methods.

Metamodels are well suited for approximating individual component models with continuous variables or responses, but they are not readily applicable to discrete or discontinuous systems. Therefore, the second task performed in this research is to

develop the CGS method for performing design optimization and design space exploration on computationally expensive, discrete variable, discontinuous response problems. The CGS method uses a Bayesian classifier in place of a metamodel to provide estimates of system performance. The classifier assigns each candidate design a categorical class label, and these class labels are used to guide the search process towards combinations of design variables that have high probabilities of yielding performance improvements. The method is compared to genetic algorithms and random search using three discrete variable optimization problems. The results of this task show that the CGS method is very effective when applied to discrete variable, discontinuous response problems. When compared to genetic algorithms, the CGS method converges to known global optima with significantly fewer function evaluations. Furthermore, the GGS method is significantly more robust than the genetic algorithm in all test cases. The CGS method consistently identifies the known global optima of the test functions across 50 trial applications for each test problem, whereas the genetic algorithm is shown to be prone to fixation on local, suboptimal solutions.

Many system-level modeling frameworks are not well suited for rapid design space exploration or optimization because each new candidate configuration requires significant modeling effort from the user; investigation of a new system architecture requires a new model to be constructed. Therefore, the third task of this research is to develop a parameterized, highly reconfigurable, system-model that can interface with the CGS algorithm. A model is developed in which the sizes, quantities and locations of energy system components are represented as vectors of discrete variables that are easily manipulated by the user or automated search algorithms. As an input, the model accepts a candidate design in the form of a string of discrete variables to represent the locations, sizes, and connectivity of energy system components. Time-dependent load profiles and

damage scenarios are used as inputs to capture the dynamic nature of notional real-life situations. The result is a tool for determining the optimal size, location, and distribution architecture of thermal and electrical system components. The tool is easily interfaced with discrete variable optimization algorithms such as the CGS method.

The fourth and final task of this research is to synthesize the first three tasks to form an early-stage design tool for determining the optimal size, location, and distribution architecture of thermal and electrical system components. Metamodels are used to approximate individual components with continuous variables and responses, and the CGS algorithm is used to explore candidate designs of system-level architectures. The highly reconfigurable energy system model is used to evaluate objective functions such as fuel consumption and survivability. For the fuel consumption study, average annual fuel consumption is calculated by using loads that represent a full spectrum of AES operating modes. The loads are weighted by the anticipated time spent in each operating mode. Notional damage scenarios are imposed on the model to assess the effects of ship configurations on survivability from an energy availability standpoint. Lastly, design tradeoffs among competing objective functions are examined. Several configurations are identified that perform well for each of the objective functions. However, an investigation of the tradeoffs among competing configurations indicates that it is possible to optimize the energy system configuration for extreme survivability without significantly reducing fuel consumption and robust survivability performance.

7.2 RESEARCH CONTRIBUTIONS

While the focus of this research is on early-stage design of AES thermal and electrical systems, the methods developed here are applicable to broad classes of

problems both inside and outside the ship design community. In this section, the contributions of this research are discussed.

7.2.1 Contribution 1: The Classifier-Guided Sampling Method

The classifier-guided sampling method is developed as a tool to perform design optimization and design space exploration on computationally expensive problems with discrete variables and discontinuous responses. Metamodels have been studied extensively as a way to alleviate computational expense, but a major shortcoming is that they are not readily applicable to discrete systems. The CGS method fills a significant void in metamodeling research. Unlike a metamodel, the classifier in the CGS algorithm does not provide estimates of the response on a continuous performance scale. However, the categorical outputs of the classifier enable it to be used as part of a direct sampling metamodel-based design optimization strategy. The classifier is inexpensive to evaluate, and it saves time by avoiding numerous evaluations of expensive objective function simulations.

As an optimization technique, the CGS method is very user-friendly compared to genetic algorithms. The effectiveness of many optimization techniques, such as genetic algorithms and tabu search, is very sensitive to the manner in which the user sets up the algorithm to solve a particular problem. Therefore, these methods are difficult to incorporate in commercial optimization software and are most appropriate for experienced users only. On the other hand, setting up the CGS method for a particular problem is fairly straightforward. There are no difficult encodings of solutions to master, and constraints are handled by simply assigning infeasible solutions a “low quality” class label. Other user-defined parameters, such as the number of expensive points to sample on each iteration and the initial proportion of “high quality” solutions to sample, have an

effect on how quickly the CGS method converges on an optimal or near optimal solution. However, if these parameters are not set perfectly, the CGS methods still performs quite well and converges to global optima in most cases. One aspect of the CGS algorithm that does require significant expertise is the structure of the Bayesian network that is used in the classifier. However, several studies suggest that the Naïve Bayes classifier performs equally well or better than more sophisticated networks in most cases [108,109].

In Chapter 4, the CGS method is tested on three discrete variable test problems: a 20-item knapsack problem, an 11-city traveling salesperson problem, and a welded beam design problem. Results show that the CGS method significantly improves the rate of convergence towards known optima when compared to GAs and random search. Although the CGS method relies on random number generation to perform some of its steps, it performs consistently well across all solution trials and is significantly less prone to fixation on local minima than GAs.

7.2.2 Contribution 2: Highly Reconfigurable Energy System Model

In Chapter 5, a highly reconfigurable model of the shipboard energy distribution system is constructed to perform design and reconfiguration studies. Typical system modeling software allows "one at a time" model simulations. That is, a specific system is "drawn" using the user interface, a simulation is run, and the result is presented to the user. If the user wants to evaluate a different architecture or configuration, a different system must be drawn and another simulation is run. The multi-energy domain, zonal modeling approach developed for this research is a highly reconfigurable system-level model, making it very appropriate for rapid design exploration and reconfiguration. In this research, MATLAB and Simulink are used for all component and system-level models and design tools. However, the models and tools are suitable for subsequent

implementation in the dynamic thermal modeling and simulation (DTMS) framework. DTMS is continuing to be developed for the specific purposes of modeling and simulating AES thermal/electrical systems, and incorporation of the modeling approach developed in this research could expand the capabilities of DTMS or other system-level modeling platforms for rapid design and reconfiguration studies.

7.2.3 Contribution 3: Framework for Complex System Design

In Chapter 6 of this dissertation, metamodels, the CGS method, and the reconfigurable energy system model are synthesized to perform design tasks on shipboard energy systems. This process composes a broadly applicable framework for designing complex, multidisciplinary engineering systems. The design framework is a compilation of the three key components. First, metamodels are used as a means to mitigate computational expense and bring previously existing non-compatible computer models into a common simulation environment. Metamodels are not a required component of the framework, but may be necessary if designers desire to incorporate existing high fidelity models in a large-scale system representation. Second, a highly reconfigurable model is used that enables the user to define a system-level configuration by tuning the values in vectors of discrete design variables. Representing a system design in this way enables users to explore competing designs rapidly by manipulating configuration variables. The last characteristic of the system-level design framework is the utilization of a discrete variable optimization algorithm to pursue solutions that improve specific operational performance metrics. The CGS method is developed and used for this purpose in this research. In general, other discrete variable optimization algorithms may also be appropriate to perform this task. These three components, when brought together effectively, form a complete framework for computer aided design of

complex engineering systems. In addition to shipboard energy systems, this approach could be applied to designing other complex, multi-disciplinary engineering systems such as aircraft, automobiles, and electrical grids that draw from a variety of conventional and renewable energy sources.

7.3 RECOMMENDATIONS FOR FUTURE WORK

There are several interesting opportunities for future work that are based on this research. The recommendations for future work fall under three categories: metamodeling, classifier-guided sampling, and electric ship design.

7.3.1 Metamodeling

The usefulness and accuracy of a metamodel is highly dependent on several method-specific user-defined tuning parameters. In this research, these parameters are tuned manually by incrementally increasing or decreasing them to reduce the number of training points required to achieve the required relative average absolute error, until a minimum number of training points is achieved. This task was only possible because the functions being approximated are inexpensive to evaluate. In practical situations when an expensive base model must be approximated, it would be extremely valuable to know precisely how to set these user-defined parameters *a priori*. Therefore, a set of reliable guidelines for setting these parameters would enable users of all experience levels to use these techniques with confidence.

7.3.2 Classifier-Guided Sampling

The classifier-guided sampling method that is presented here is applied to problems with relatively small design spaces that range from approximately 37,000 possible solutions for the AES fuel consumption study to approximately 1.8 million possible solutions for the 11-city traveling salesperson problem. For each iteration of the

CGS algorithm, the classifier assigns categorical class labels to *all* possible solutions when the problems are this size. However, many engineering problems have design spaces that are so large that classification of the entire space is prohibitively time consuming. For example, a 25 city traveling salesperson problem has 24!/2 (approximately 3.1×10^{23}) possible solutions. Addressing this issue would enable the CGS method to be applied to a much broader range of problem types.

The strategy for setting and changing the class threshold in the CGS algorithm has a significant effect on performance. The class threshold is used to assign class labels to the training points at each iteration. If the threshold is too stringent, very few of the training points are labeled as 'high' quality, and the classifier does not get a sample of high quality points that is large enough for it to learn the characteristics of high quality solutions. On the other hand, if the threshold is not stringent enough, the classifier is trained with a large number of points with the 'high' quality class label, many of which do not actually result in strong objective function performance. In this case, the classifier subsequently predicts that a large number of the cheap points are high quality solutions, and the probability that the true high quality points are sampled for expensive evaluation is significantly reduced. The strategy for changing the class thresholds that is implemented in this research is effective but simple, and a more sophisticated algorithm for changing these thresholds may improve the ability of the CGS to converge quickly to good solutions.

Choosing an appropriate Bayesian network structure for the classifier is also critical. In this research, the fully independent, or Naïve Bayes, structure is used for all problems except for the traveling salesperson problem. When a Naïve Bayes structure is used, the dimensionalities of each variable's class conditional probabilities are low, and fewer training points are required to populate the distribution. For many problems, a

more accurate classifier will result if a more sophisticated Bayesian network classifier is used. A careful understanding of the problem and the conditional dependencies of the variables involved could be used to choose a more appropriate Bayesian network structure and improve the ability of the classifier to predict solution classes accurately.

The CGS algorithm has three user-defined constants: the number of training points, the number of expensive points to sample at each iteration, and the initial proportion of high-certainty high-class points to sample. For the validation studies that are presented in Chapter 4, these parameters are tuned in an *ad hoc* manner by testing all combinations of three settings of each parameter, for 27 total combinations. The best combinations of these settings is different for all three problems which indicates that the ideal values for these parameters depends on the nature of the problem being solved. In many situations, computational expense and lack of knowledge of the objective function global optima make this approach to parameter tuning impractical. Therefore, an investigation into the effects of these parameters would yield useful insights and a better understanding about how they should be set depending on the problem to which the CGS method is being applied.

7.3.3 All-Electric Ship Design

The reconfigurable AES model that is developed here lacks the detail of a high-fidelity, physics-based model that could be developed in more sophisticated modeling frameworks such as DTMS. Furthermore, notional load scenarios are developed using approximate engineering equations and limited publicly available specifications. Therefore, the concepts suggested by the configuration studies in this research are not ready to be directly implemented in the final design of a ship. However, the solutions found are strong candidates for more detailed investigation. A more sophisticated model

with real naval load data could be used as part of an iterative design process to validate and verify the performance of these candidate configurations.

Appendix A: Welded Beam Problem Constants and Constraints

The following constants and constraint equations correspond to the welded beam design optimization problem that is explained in Section 4.3.

MATERIAL PROPERTIES AND COSTS

Table A.1: Welded beam problem material properties and costs [103]

Material	*S* (*kpsi*)	*E* (*Mpsi*)	*G* (*Mpsi*)	c_1 (*$/in.*^{3})	c_2 (*$/in.*^{3})
Steel	30	30	12	0.1047	0.0481
Cast Iron	8	14	6	0.0489	0.0224
Aluminum	5	10	4	0.5235	0.2405
Brass	8	16	6	0.5584	0.2566

In Table A.1, S is the design stress, E is the Young's modulus, and G is the shearing modulus.

CONSTRAINT EQUATIONS

The following set of equations comprises the bending stress, buckling load, beam deflection, and shear stress constraint equations for the welded beam problem. In the following equations, the x_i's represent the six design parameters that are explained in Section 4.3. L is the extended (non-welded) length of the beam, and F is the force on the end of the beam.

Constraint g_1: *Beam bending stress* $\sigma(\mathbf{x})$. The beam bending stress is equal to:

$$\sigma(\mathbf{x}) = \frac{6FL}{x_5 x_4^2} \tag{A.1}$$

Constraint g_2: *Beam buckling load* $P_c(\mathbf{x})$. The beam buckling load approximated by

$$P_c(\mathbf{x}) = \frac{4.013\sqrt{EI\alpha}}{L^2}\left[1 - \frac{x_4}{2L}\sqrt{\frac{EI}{\alpha}}\right] \tag{A.2}$$

The second moment of area I in Equation (A.2) is given by

$$I = \frac{x_4 x_5^3}{12} \tag{A.3}$$

and the constant α in Equation (A.2) is given by

$$\alpha = \frac{G x_4 x_5^3}{3} \tag{A.4}$$

Constraint g_3: *Beam deflection* $\delta(\mathbf{x})$. The beam deflection is calculated by modeling the bar as a cantilever of length L:

$$\delta(\mathbf{x}) = \frac{4FL^3}{E x_5 x_4^3} \tag{A.5}$$

Constraint g_4: *Weld shear stress* $\tau(\mathbf{x})$. The weld shear stress has two components, τ_1 and τ_2. The primary shear stress is given by

$$\tau_1 = \frac{F}{A} \tag{A.6}$$

where the area, A, is given by

$$A = \begin{cases} 1.414 x_3 x_6, & x_1 = 0 \\ 1.414 x_3 (x_4 + x_6), & x_1 = 1 \end{cases} \tag{A.5}$$

The secondary shear stress is given by

$$\tau_2 = \frac{MR}{J} \tag{A.8}$$

The bending moment, M, in Equation (A.8) is given by

$$M = F\left[L + \frac{x_6}{2} \right] \tag{A.9}$$

The torsion radius, R, in Equation (A.8) is given by

$$R = \left\{ \frac{x_6^2}{4} + \left[\frac{x_3 + x_4}{2} \right]^2 \right\}^{1/2} \tag{A.10}$$

Lastly, the polar moment of inertia, J, in Equation (A.8) is given by

$$J = \begin{cases} 1.414 x_3 x_6 \left[\dfrac{(x_3 + x_4)^2}{4} + \dfrac{x_6^2}{12} \right], & x_1 = 0 \\ 1.414 x_3 \dfrac{(x_3 + x_4 + x_6)^3}{12}, & x_1 = 1 \end{cases} \tag{A.11}$$

The primary and secondary shear stresses combine to form the total weld shear stress as follows:

$$\tau(\mathbf{x}) = \left[\tau_1^2 + \frac{x_6}{R} \tau_1 \tau_2 + \tau_2^2 \right]^{1/2} \tag{A.12}$$

Appendix B: All-Electric Ship Volumetric Constraint

Equipment space is limited on Navy ships, and volumetric constraints must be considered when evaluating candidate equipment configurations. The methodology for developing and calculating the zonal volumetric constraints is presented here. In Table B.1, the volumes of available design components are given. These specifications are obtained from technical reports, journal publications, and manufacturer marketing literature.

Table B.1: Energy production and storage equipment volumes

Component	*Volume (ft^3)*	*Source*
MT30 Gas Turbine Generator	**11,137**	**[117]**
RR4500 Gas Turbine Generator	**4,040**	**[118]**
Chilled Water Plant	**394**	**[119]**
Fan Coil Assembly	**11**	**[110]**
Cooling Coil – Size 61	3.1	p. 34
Cooling Coil Fan – NS Centrifugal CC1/2	7.9	p.27
Flywheel	**345**	**[116]**

The first step in estimating the available space for equipment is to calculate the total volume of components necessary to meet anticipated ship loads. To satisfy the load scenarios that are outlined in Chapter 6, two RR4500s, two MT30s, thirteen chilled water plants, and 485 cooling coils are necessary. Although energy storage components may improve ship survivability, they are not required to satisfy the steady state loads. Therefore, flywheels are not included in the calculation of total available equipment space. The volume of all required equipment (generators, chillers, and cooling coils) is multiplied by 1.75 to provide additional flexibility for zonal placement of components, resulting in an available equipment space of 71,419 ft^3. The available equipment space in each zone is calculated by assuming each zone has its own percentage of total available equipment space, as shown in Table B.2:

Table B.2: Zonal volume percentage of total volume available

	Zone 1	*Zone 2*	*Zone 3*	*Zone 4*	*Zone 5*	*Total*
Percentage	10%	33%	35%	15%	7%	100%
Volume (ft^3)	7,142	23,568	24,997	10,713	4,999	71,419

The generators and chilled water plants can conceivably be placed anywhere on the ship because the commodities they produce can be transported through pipes and electrical busses. However, all refrigerated air is produced locally, and a set number of cooling coils must be placed in each zone according to zonal refrigerated air loads. Table B.3 shows the number of fan coil assemblies that are needed in each zone to satisfy refrigerated air demand.

Table B.3: Required cooling coils by zone

	Zone 1	*Zone 2*	*Zone 3*	*Zone 4*	*Zone 5*	*Total*
Fan Coils	26	47	35	17	360	485
Volume (ft^3)	286	517	385	187	3,960	5,335

After subtracting the volume of required cooling coils, the total volume available for generators, chilled water plants, and flywheels in each zone is shown in Table B.4:

Table B.4: Volume available for design variable components

	Zone 1	*Zone 2*	*Zone 3*	*Zone 4*	*Zone 5*	*Total*
Volume (ft^3)	6,856	23,051	24,612	10,526	1,039	66,084

The zone-specific volumes in Table B.4 are used in the optimization studies discussed in Chapter 6 to determine whether a candidate configuration violates volumetric constraints.

Appendix C: Best AES Configurations Identified with CGS

The AES configurations presented in this appendix are found by applying the CGS method to the three objective functions that are described in Chapter 6. The solutions presented here for each objective function are the five best *unique* solutions obtained. However, each run of the CGS method does not necessarily produce a unique result. Therefore, a configuration presented here may the second or third best configuration obtained by one of the five repeat runs of the CGS method.

FUEL CONSUMPTION OBJECTIVE FUNCTION BEST CONFIGURATIONS

Table C.1: Fuel consumption objective configuration 1

Category	*Component / Mode*	*Z1*	*Z2*	*Z3*	*Z4*	*Z5*
Configuration Variables	RR4500s	1	0	0	1	0
	MT30s	0	0	2	0	0
	CW Plants	2	0	1	8	2
Electrical ZSD Topology	Shore	1	1	1	1	1
	Anchor	1	1	1	1	1
	Low Speed	1	1	1	1	1
	Cruise	1	1	1	1	1
	Sustained Battle	1	1	1	1	1
	Full Speed	1	0	1	1	0
Chilled Water ZSD Topology	Shore	1	1	0	1	1
	Anchor	1	1	0	1	1
	Low Speed	1	0	1	0	1
	Cruise	1	0	1	0	1
	Sustained Battle	1	1	1	1	1
	Full Speed	1	0	1	0	1
Objective Function Value						
Average rate of fuel consumption (bbl/yr.):				191,843.6		

Table C.2: Fuel consumption objective configuration 2

Category	*Component / Mode*	*Z1*	*Z2*	*Z3*	*Z4*	*Z5*
Configuration Variables	RR4500s	0	1	0	1	0
	MT30s	0	0	2	0	0
	CW Plants	8	0	1	2	2
Electrical ZSD Topology	Shore	1	1	1	1	1
	Anchor	1	1	1	1	1
	Low Speed	1	1	1	1	1
	Cruise	1	1	1	1	1
	Sustained Battle	1	1	1	1	1
	Full Speed	1	0	1	1	0
Chilled Water ZSD Topology	Shore	0	1	1	1	1
	Anchor	0	1	1	1	1
	Low Speed	1	0	1	0	1
	Cruise	1	0	1	0	1
	Sustained Battle	1	1	1	1	1
	Full Speed	1	0	1	0	1
Objective Function Value						
Average rate of fuel consumption (bbl/yr.):				191,850.9		

Table C.3: Fuel consumption objective configuration 3

Category	*Component / Mode*	*Z1*	*Z2*	*Z3*	*Z4*	*Z5*
Configuration Variables	RR4500s	1	0	0	1	0
	MT30s	0	0	2	0	0
	CW Plants	2	1	0	8	2
Electrical ZSD Topology	Shore	1	1	1	1	1
	Anchor	1	1	1	1	1
	Low Speed	1	1	1	1	1
	Cruise	1	1	1	1	1
	Sustained Battle	1	1	1	1	1
	Full Speed	1	0	1	1	0
Chilled Water ZSD Topology	Shore	1	1	0	1	1
	Anchor	1	1	0	1	1
	Low Speed	1	0	1	0	1
	Cruise	1	0	1	0	1
	Sustained Battle	1	1	1	1	1
	Full Speed	1	0	1	0	1
Objective Function Value						
Average rate of fuel consumption (bbl/yr.):				191,868.1		

Table C.4: Fuel consumption objective configuration 4

Category	*Component / Mode*	*Z1*	*Z2*	*Z3*	*Z4*	*Z5*
Configuration Variables	RR4500s	1	0	0	1	0
	MT30s	0	0	2	0	0
	CW Plants	5	1	0	5	2
Electrical ZSD Topology	Shore	1	1	1	1	1
	Anchor	1	1	1	1	1
	Low Speed	1	1	1	1	1
	Cruise	1	1	1	1	1
	Sustained Battle	1	1	1	1	1
	Full Speed	1	0	1	1	0
Chilled Water ZSD Topology	Shore	1	1	1	1	1
	Anchor	1	1	1	1	1
	Low Speed	1	1	1	0	0
	Cruise	1	1	1	0	0
	Sustained Battle	1	1	1	1	1
	Full Speed	1	1	1	0	0
Objective Function Value						
Average rate of fuel consumption (bbl/yr.):				191,872.1		

Table C.5: Fuel consumption objective configuration 5

Category	*Component / Mode*	*Z1*	*Z2*	*Z3*	*Z4*	*Z5*
Configuration Variables	RR4500s	0	1	0	1	0
	MT30s	0	0	2	0	0
	CW Plants	6	3	0	2	2
Electrical ZSD Topology	Shore	1	1	1	1	1
	Anchor	1	1	1	1	1
	Low Speed	1	1	1	1	1
	Cruise	1	1	1	1	1
	Sustained Battle	1	1	1	1	1
	Full Speed	1	0	1	1	0
Chilled Water ZSD Topology	Shore	1	1	1	1	1
	Anchor	1	1	1	1	1
	Low Speed	0	1	1	1	1
	Cruise	0	1	1	1	1
	Sustained Battle	1	1	1	1	1
	Full Speed	0	1	1	1	1
Objective Function Value						
Average rate of fuel consumption (bbl/yr.):				191,998.2		

EXTREME SURVIVABILITY OBJECTIVE FUNCTION BEST CONFIGURATIONS

Table C.6: Extreme survivability objective configuration 1

Category	*Component / Damage*	*Z1*	*Z2*	*Z3*	*Z4*	*Z5*
Configuration Variables	RR4500s	0	2	0	0	0
	MT30s	0	0	2	0	0
	CW Plants	10	1	0	1	1
	Flywheels	8	0	0	0	0
Electrical Unmet Demand (kJ)	Damage Scenario 1	1,207	0.3	89.9	0.1	2.5
	Damage Scenario 2	1,062	0.2	95.6	0.2	2.7
	Damage Scenario 3	0.6	0.4	148.1	14.9	4.0
	Damage Scenario 4	0.3	0.2	77.9	12.8	2.1
*Heat Dissipation Unmet Demand (kJ)	Damage Scenario 1	312.5	0	0	0	0
	Damage Scenario 2	20.2	0.9	1.8	0.3	6.9
	Damage Scenario 3	17.1	0.3	0.6	0	2.4
	Damage Scenario 4	17.1	0.3	0.6	7.9	2.4
Objective Function Value						
Maximum Total Unmet Demand (kJ):				1,612.1		

*Damage scenarios do not impact cooling coil availability, and the model calculates zero unmet demand for all zonal refrigerated air loads. In reality, insufficient chilled water supply results in refrigerated air unmet demand, because each available online cooling coil requires chilled water to operate. In this research, the refrigerated air unmet demand that results from inadequate chilled water supply is accounted for in the chilled water unmet demand. Therefore, the chilled water and refrigerated air unmet demands are summed into one measure of thermal heat dissipation unmet demand.

Table C.7: Extreme survivability objective configuration 2

Category	*Component / Damage*	*Z1*	*Z2*	*Z3*	*Z4*	*Z5*
Configuration Variables	RR4500s	0	2	0	0	0
	MT30s	0	0	2	0	0
	CW Plants	10	1	2	0	0
	Flywheels	8	0	0	0	0
Electrical Unmet Demand (kJ)	Damage Scenario 1	1,207	0.3	90	0.1	2.5
	Damage Scenario 2	1,062	0.2	95.8	0.1	2.6
	Damage Scenario 3	0.6	0.4	148.4	12.8	4.0
	Damage Scenario 4	0.2	0.2	77.8	12.8	2.1
*Heat Dissipation Unmet Demand (kJ)	Damage Scenario 1	312.5	0	0	0	0
	Damage Scenario 2	20.2	0.9	1.8	0.3	6.9
	Damage Scenario 3	17.1	0.3	0.6	7.9	2.4
	Damage Scenario 4	0	0	0	7.9	0
Objective Function Value						
Maximum Total Unmet Demand (kJ):				1,612.1		

Table C.8: Extreme survivability objective configuration 3

Category	*Component / Damage*	*Z1*	*Z2*	*Z3*	*Z4*	*Z5*
Configuration Variables	RR4500s	0	1	0	1	0
	MT30s	0	0	2	0	0
	CW Plants	10	1	0	2	0
	Flywheels	8	0	0	0	0
Electrical Unmet Demand (kJ)	Damage Scenario 1	1,207	0.3	89.9	0.2	2.4
	Damage Scenario 2	1,062	0.2	95.6	0.3	2.6
	Damage Scenario 3	0.6	0.4	165.6	0	4.5
	Damage Scenario 4	0.3	0.2	83.3	14.9	2.2
*Heat Dissipation Unmet Demand (kJ)	Damage Scenario 1	312.5	0	0	0	0
	Damage Scenario 2	20.2	0.9	1.8	0.3	6.9
	Damage Scenario 3	66.1	1.26	2.4	0	9.2
	Damage Scenario 4	66.1	1.26	2.4	0	9.2
Objective Function Value						
Maximum Total Unmet Demand (kJ):				1,612.1		

Table C.9: Extreme survivability objective configuration 4

Category	*Component / Damage*	*Z1*	*Z2*	*Z3*	*Z4*	*Z5*
Configuration Variables	RR4500s	0	1	0	1	0
	MT30s	0	0	2	0	0
	CW Plants	10	1	0	1	1
	Flywheels	8	0	0	0	0
Electrical Unmet Demand (kJ)	Damage Scenario 1	1,207	0.3	89.9	0.1	2.5
	Damage Scenario 2	1,062	0.2	95.6	0.2	2.7
	Damage Scenario 3	0.6	0.4	165.6	0	4.5
	Damage Scenario 4	0.3	0.2	83.3	12.8	2.2
*Heat Dissipation Unmet Demand (kJ)	Damage Scenario 1	312.5	0	0	0	0
	Damage Scenario 2	20.2	0.9	1.8	0.3	6.9
	Damage Scenario 3	17.1	0.3	0.6	0	2.4
	Damage Scenario 4	17.1	0.3	0.6	7.9	2.4
Objective Function Value						
Maximum Total Unmet Demand (kJ):				1,612.1		

Table C.10: Extreme survivability objective configuration 5

Category	*Component / Damage*	*Z1*	*Z2*	*Z3*	*Z4*	*Z5*
Configuration Variables	RR4500s	0	0	0	2	0
	MT30s	0	0	2	0	0
	CW Plants	10	1	0	1	1
	Flywheels	8	0	0	0	0
Electrical Unmet Demand (kJ)	Damage Scenario 1	1,207	0.3	89.9	0.1	2.5
	Damage Scenario 2	1,062	0.2	89.9	0.2	2.5
	Damage Scenario 3	0.7	0.5	187.5	0	5.1
	Damage Scenario 4	0.3	0.2	88.9	0	2.4
*Heat Dissipation Unmet Demand (kJ)	Damage Scenario 1	312.5	0	0	0	0
	Damage Scenario 2	20.2	0.9	1.8	0.3	6.9
	Damage Scenario 3	17.1	0.3	0.6	0	2.4
	Damage Scenario 4	17.1	0.3	0.6	7.9	2.4
Objective Function Value						
Maximum Total Unmet Demand (kJ):				1,612.1		

ROBUST SURVIVABILITY OBJECTIVE FUNCTION BEST CONFIGURATIONS

Table C.11: Robust survivability objective configuration 1

Category	*Component / Damage*	*Z1*	*Z2*	*Z3*	*Z4*	*Z5*
Configuration Variables	RR4500s	1	1	0	0	0
	MT30s	0	0	2	0	0
	CW Plants	7	6	0	0	0
	Flywheels	0	0	0	6	2
Electrical Unmet Demand (kJ)	Damage Scenario 1	1,376	0.4	83.2	0.1	2.2
	Damage Scenario 2	85.2	0.4	88.9	0.1	2.4
	Damage Scenario 3	0.5	0.5	164.2	0	4.4
	Damage Scenario 4	0.2	0.3	86.9	0	2.3
*Heat Dissipation Unmet Demand (kJ)	Damage Scenario 1	429.7	0	0	0	0
	Damage Scenario 2	195.5	0	0	0	0
	Damage Scenario 3	0	0	0	7.9	0
	Damage Scenario 4	0	0	0	7.9	0
Objective Function Value						
Mean Plus Std. Dev. of Total Unmet Demand (kJ):				1,480.3		

*Damage scenarios do not impact cooling coil availability, and the model calculates zero unmet demand for all zonal refrigerated air loads. In reality, insufficient chilled water supply results in refrigerated air unmet demand, because each available online cooling coil requires chilled water to operate. In this research, the refrigerated air unmet demand that results from inadequate chilled water supply is accounted for in the chilled water unmet demand. Therefore, the chilled water and refrigerated air unmet demands are summed into one measure of thermal heat dissipation unmet demand.

Table C.12: Robust survivability objective configuration 2

Category	*Component / Damage*	*Z1*	*Z2*	*Z3*	*Z4*	*Z5*
Configuration Variables	RR4500s	1	1	0	0	0
	MT30s	0	0	2	0	0
	CW Plants	7	5	0	0	1
	Flywheels	0	0	0	8	0
Electrical Unmet Demand (kJ)	Damage Scenario 1	1,376	0.3	83.1	0.1	2.2
	Damage Scenario 2	85.2	0.3	88.9	0.1	2.4
	Damage Scenario 3	0.5	0.5	169.9	0	4.6
	Damage Scenario 4	0.2	0.3	89.9	0	2.4
*Heat Dissipation Unmet Demand (kJ)	Damage Scenario 1	429.7	0	0	0	0
	Damage Scenario 2	195.5	0	0	0	0
	Damage Scenario 3	0	0	0	7.9	0
	Damage Scenario 4	0	0	0	7.9	0
Objective Function Value						
Mean Plus Std. Dev. of Total Unmet Demand (kJ):				1,480.7		

Table C.13: Robust survivability objective configuration 3

Category	*Component / Damage*	*Z1*	*Z2*	*Z3*	*Z4*	*Z5*
Configuration Variables	RR4500s	1	1	0	0	0
	MT30s	0	0	2	0	0
	CW Plants	7	4	0	0	2
	Flywheels	0	0	0	8	0
Electrical Unmet Demand (kJ)	Damage Scenario 1	1,376	0.3	83.2	0.1	2.3
	Damage Scenario 2	85.2	0.3	88.8	0.1	2.4
	Damage Scenario 3	0.5	0.5	169.9	0	4.6
	Damage Scenario 4	0.2	0.3	89.9	0	2.4
*Heat Dissipation Unmet Demand (kJ)	Damage Scenario 1	429.7	0	0	0	0
	Damage Scenario 2	195.5	0	0	0	0
	Damage Scenario 3	0	0	0	7.9	0
	Damage Scenario 4	0	0	0	7.9	0
Objective Function Value						
Mean Plus Std. Dev. of Total Unmet Demand (kJ):				1,480.8		

Table C.14: Robust survivability objective configuration 4

Category	*Component / Damage*	*Z1*	*Z2*	*Z3*	*Z4*	*Z5*
Configuration Variables	RR4500s	1	1	0	0	0
	MT30s	0	0	2	0	0
	CW Plants	7	6	0	0	0
	Flywheels	0	0	0	8	0
Electrical Unmet Demand (kJ)	Damage Scenario 1	1,376	0.4	83.2	0.1	2.2
	Damage Scenario 2	85.2	0.4	88.9	0.1	2.4
	Damage Scenario 3	0.5	0.5	169.9	0	4.6
	Damage Scenario 4	0.2	0.3	89.9	0	2.4
*Heat Dissipation Unmet Demand (kJ)	Damage Scenario 1	429.7	0	0	0	0
	Damage Scenario 2	195.5	0	0	0	0
	Damage Scenario 3	0	0	0	7.9	0
	Damage Scenario 4	0	0	0	7.9	0
Objective Function Value						
Mean Plus Std. Dev. of Total Unmet Demand (kJ):				1,480.3		

Table C.15: Robust survivability objective configuration 5

Category	*Component / Damage*	*Z1*	*Z2*	*Z3*	*Z4*	*Z5*
Configuration Variables	RR4500s	1	0	0	1	0
	MT30s	0	0	2	0	0
	CW Plants	7	5	0	0	1
	Flywheels	0	0	0	8	0
Electrical Unmet Demand (kJ)	Damage Scenario 1	1,376	0.3	83.1	0.1	2.2
	Damage Scenario 2	85.2	0.3	83.2	0.1	2.3
	Damage Scenario 3	0.6	0.6	189.9	0	5.1
	Damage Scenario 4	0.3	0.3	95.7	0	2.6
*Heat Dissipation Unmet Demand (kJ)	Damage Scenario 1	429.7	0	0	0	0
	Damage Scenario 2	195.5	0	0	0	0
	Damage Scenario 3	0	0	0	7.9	0
	Damage Scenario 4	0	0	0	7.9	0
Objective Function Value						
Mean Plus Std. Dev. of Total Unmet Demand (kJ):				1,481.6		

References

[1] ESRDC, *Electric Ship Research and Development Consortium,* http://www.esrdc.com/. Date: October 14th, 2011

[2] Doerry, N. H. and J. C. Davis, 1994, "Integrated Power System for Marine Applications," *Naval Engineers Journal,* Vol. 106, No. 3, pp. 77-90.

[3] Doerry, N., H. Robey, J. Amy and C. Petry, 1996, "Powering the Future with the Integrated Power System," *Naval Engineers Journal,* Vol. 108, No. 3, pp. 267-282.

[4] Zerby, M., 2006, "Thermal Management for the Electric Warship," *ASNE Electric Machines Technology Symposium*, Philadelphia, PA, May 22-24, 2006.

[5] Kiehne, T. M., 2011, "Dynamic Assessment of Thermal Management Strategies aboard Naval Surface Ships," *IEEE Electric Ship Technologies Symposium (ESTS 2011)*, Alexandria, VA, April 10-13, 2011.

[6] Doerry, N. H. and H. Fireman, 2006, "Designing All Electric Ships," *Proceedings of the Ninth International Marine Design Conference*, Ann Arbor, MI.

[7] Butler, K. L. and N. D. R. Sarma, 2000, "General Reconfiguration Methodology for AC Radial Shipboard Power Systems," *IEEE Power Engineering Society Meeting Winter Meeting,* Vol. 2, pp. 1226-1230.

[8] Petry, C. R. and J. W. Rumburg, 1993, "Zonal Electrical Distribution Systems: An Affordable Architecture for the Future," *Naval Engineers Journal,* Vol. 105, No. 3, pp. 45-51.

[9] Shiffler, M. E., 1993, "Development of a Zonal Architecture Fire Main System for Combatant Ships," *Naval Engineers Journal,* Vol. 105, No. 3, pp. 31-44.

[10] Doerry, N. H., 2006, "Zonal Ship Design," *Naval Engineers Journal,* Vol. 118, No. 1, pp. 39-53.

[11] Doerry, N. H., 2007, "Designing Electric Power Systems for Survivability and Quality of Service," *Naval Engineers Journal,* Vol. 119, No. 2, pp. 25-34.

[12] Chalfant, J. S. and C. Chryssostomidis, 2011, "Analysis of Various All-Electric-Ship Electrical Distribution System Topologies," *IEEE Electric Ship Technologies Symposium (ESTS 2011)*, Alexandria, VA, April 11-13, 2011.

[13] Syntek "DD(X) Notional Baseline Modeling and Simulation Development Report in Support of the Electric Ship Research and Development Consortium", Intra-Consortium Report, 1 August 2003.

[14] Nadeem, M., 2001, "Evaluation of Overall Chiller Performance Characteristics". *Air Conditioning and Refrigeration Journal - The Magazine of the Indian Society of Heating, Refrigeration and Air Conditioning Engineers*, July-September 2001.

[15] Sobieszczanski-Sobieski, J. and R. T. Haftka, 1997, "Multidisciplinary Aerospace Design Optimization: Survey of Recent Developments," *Structural Optimization,* Vol. 14, pp. 1-23.

[16] Sobieszczanski-Sobieski, J., J. S. Agte and R. R. Sanduski "Bi-Level Integrated System Synthesis (BLISS)", *Tech. Report NASA / TM-1998-208715*, National Aeronautics and Space Administration, August 1998.

[17] Simpson, T. W., J. D. Peplinski, P. N. Koch and J. K. Allen, 2001, "Metamodels for Computer-based Engineering Design: Survey and Recommendations," *Engineering with Computers,* Vol. 17, pp. 129-150.

[18] Shan, S. and G. G. Wang, 2010, "Survey of Modeling and Optimization Strategies to Solve High-Dimensional Design Problems with Computationally-Expensive Black-Box functions," *Structural and Multidisciplinary Optimization,* Vol. 41, pp. 219-241.

[19] Wang, G. G. and S. Shan, 2007, "Review of Metamodeling Techniques in Support of Engineering Design Optimization," *Transactions of the ASME,* Vol. 129, pp. 370-380.

[20] Simpson, T. W., V. Toropov, V. Balabanov and F. A. C. Viana, 2008, "Design and Analysis of Computer Experiments in Multidisciplinary Design Optimization: A Review of How Far We Have Come – or Not," *12th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, Victoria, British Columbia, 10-12 September, 2008.

[21] Paullus, P. E., 2007, "Creation of a Modeling and Simulation Environment for Thermal Management of an All-Electric Ship," *MS Thesis,* Mechanical Engineering Department, University of Texas at Austin.

[22] Pierce, M. S., 2009, "Dynamic Thermal Modeling and Simulation Framework: Design of Modeling Techniques and External Integration Tools," *MS Thesis,* Mechanical Engineering Department, University of Texas at Austin.

[23] Pierce, M. S. and T. M. Kiehne, 2010, "DTMS: A Framework for System-Level, Dynamic Simulations across Multi-Disciplinary Boundaries," *Grand Challenges in Modeling and Simulation*, Ottawa, July, 2010.

[24] Brice, C. W., L. U. Gokdere and R. A. Dougal, 1998, "The Virtual Test Bed: An Environment for Virtual Prototyping," *Proceedings of the International Conference on Electric Ships*, Istanbul, Turkey, September 1998.

[25] Fang, R., W. Jiang, A. Monti, M. Zerby, G. Anderson, P. Bernotas and J. Khan, 2007, "System-Level Dynamic Thermal Modeling and Simulation for an All-Electric Ship Cooling System in VTB," *IEEE Electric Ship Technologies Symposium (ESTS 2007)*, Arlington, VA, May 21-23, 2007.

[26] Box, G. E. P. and K. B. Wilson, 1951, "On the Experimental Attainment of Optimal Conditions," *Journal of the Royal Statistical Society,* Vol. 13, pp. 1-45.

[27] Friedman, J. H., 1991, "Multivariate Adaptive Regression Splines," *The Annals of Statistics,* Vol. 19, No. 1, pp. 1-67.

[28] Vapnik, V., S. Golowich and A. Smola, 1997, "Support Vector Method for Function Approximation, Regression Estimation, and Signal Processing," *Advances in Neural Information Processing Systems,* Vol. 9, pp. 281-287.

[29] Sacks, J., S. B. Schiller and W. J. Welch, 1989, "Designs for Computer Experiments," *Technometrics,* Vol. 31, No. 1, pp. 41-47.

[30] Buhmann, M. D., 2003, *Radial Basis Functions: Theory and Implementations, 1st Ed.,* Cambridge University Press.

[31] Cheng, B. and D. M. Titterington, 1994, "Neural Networks: A Review from a Statistical Perspective," *Statistical Science,* Vol. 9, No. 1, pp. 2-54.

[32] Meckesheimer, M., A. J. Booker, R. R. Barton and T. Simpson, 2002, "Computationally Inexpensive Metamodel Assessment Strategies," *AIAA Journal,* Vol. 40, No. 10, pp. 2053-2060.

[33] Booker, A. J., J. E. Dennis, P. D. Frank, D. B. Serafini, V. Torczon and M. W. Trosset, 1999, "A Rigorous Framework for Optimization of Expensive Functions by Surrogates," *Structural Optimization,* Vol. 17, pp. 1-13.

[34] Dennis, J. E. and V. Torczon, 1996, *Managing Approximation Models in Optimization, 1st Ed.,* N. Alexandrov and M. Hussaini, eds., Society for Industrial and Applied Mathematics, Philadelphia.

[35] Wang, L., S. Shan and G. G. Wang, 2004, "Mode-Pursuing Sampling Method for Global Optimization on Expensive Black-Box Functions," *Engineering Optimization,* Vol. 36, No. 4, pp. 419-438.

[36] Fang, H., M. Rais-Rohani, Z. Liu and M. F. Horstemeyer, 2005, "A Comparative Study of Metamodeling Methods for Multiobjective Crashworthiness Optimization," *Computers and Structures,* Vol. 83, pp. 2121-2136.

[37] Cappelleri, D. J., M. I. Frecker and T. W. Simpson, 2002, "Design of a PZT Bimorph Actuator Using a Metamodel-Based Approach," *Transactions of the ASME,* Vol. 124, pp. 354-357.

[38] Barthelemy, J. F. M. and R. T. Haftka, 1993, "Approximation Concepts for Optimum Structural Design - A Review," *Structural Optimization,* Vol. 5, pp. 129-144.

[39] Praveen, C. and R. Duvigneau, 2009, "Low Cost PSO Using Metamodels and Inexact Pre-Evaluations: Application to Aerodynamic Shape Design," *Computer Methods in Applied Mechanics and Engineering,* Vol. 198, pp. 1087-1096.

[40] Michelena, N., H. M. Kim and P. Papalambros, 1999, "A System Partitioning and Optimization Approach to Target Cascading," *International Conference on Engineering Design (ICED '99)*, Munich, August 24-26, 1999.

[41] Batill, S. M. and M. A. Stelmack, 1999, "Framework for Multidisciplinary Design Based on Response-Surface Approximations," *Journal of Aircraft,* Vol. 36, No. 1, pp. 287-297.

[42] Giunta, A. and L. T. Watson, 1998, "A Comparison of Approximation Modeling Techniques: Polynomial Versus Interpolating Models," *7th AIAA/USAF/ISSMO Symposium on Multidisciplinary Analysis & Optimization*, St. Louis, MO, AIAA, Vol. 1, pp.392-404. AIAA-98-4758.

[43] Wang, X., Y. Liu and E. K. Antonsson, 1999, "Fitting Functions to Data in High Dimensionsal Design Space," *ASME Design Engineering Technical Conferences*, Las Vegas, NV, Paper Number DETC99/DAC-8622.

[44] Jin, R., W. Chen and T. W. Simpson, 2001, "Comparative Studies of Metamodelling Techniques Under Multiple Modelling Criteria," *Structural and Multidisciplinary Optimization,* Vol. 23, pp. 1-13.

[45] Clarke, S. M., J. H. Griebsch and T. W. Simpson, 2005, "Analysis of Support Vector Regression for Approximation of Complex Engineering Analyses," *Journal of Mechanical Design,* Vol. 127, pp. 1077-1087.

[46] Lee, Y., S. Oh and D.-H. Choi, 2008, "Design Optimization Using Support Vector Regression," *Journal of Mechanical Science and Technology,* Vol. 22, pp. 213-220.

[47] Kim, B.-S., Y.-B. Lee and D.-H. Choi, 2009, "Comparison Study on the Accuracy of Metamodeling Technique for Non-Convex Functions," *Journal of Mechanical Science and Technology,* Vol. 23, pp. 1175-1181.

[48] Ely, G. R. and C. C. Seepersad, 2009, "A Comparative Study of Metamodeling Techniques for Predictive Process Control of Welding Applications," *International Manufacturing Science and Engineering Conference*, West Lafayette, Indiana, ASME, Paper Number MSEC2009-84189.

[49] Forrester, A. I. J. and A. J. Keane, 2009, "Recent Advances in Surrogate-Based Optimization," *Progress in Aerospace Sciences,* Vol. 45, pp. 50-79.

[50] Huang, M. W. and J. S. Arora, 1997, "Optimal Design with Discrete Variables: Some Numerical Examples," *International Journal for Numerical Methods in Engineering,* Vol. 40, pp. 165-188.

[51] Meckesheimer, M., R. R. Barton, T. W. Simpson, F. Limayem and B. Yannou, 2001, "Metamodeling of Combined Discrete/Continuous Responses," *AIAA Journal,* Vol. 39, No. 10, pp. 1950-1959.

[52] Sharif, B., G. G. Wang and T. Y. ElMekkawy, 2008, "Mode Pursuing Sampling Method for Discrete Variable Optimization on Expensive Black-Box Functions," *Journal of Mechanical Design,* Vol. 130, pp. 021402-1-11.

[53] Kirkpatrick, S., C. D. Gelatt and M. P. Vecchi, 1983, "Optimization by Simulated Annealing," *Science,* Vol. 220, No. 4598, pp. 671-680.

[54] Cowan, M. J., 2008, "Breaking Short Playfair Cyphers with the Simulated Annealing Algorithm," *Cryptologia,* Vol. 32, pp. 71-83.

[55] Eglese, R. W., 1990, "Simulated Annealing: A Tool for Operational Research," *European Journal of Operational Research,* Vol. 46, pp. 271-281.

[56] Granville, V., M. Krivanek and J. P. Rasson, 1994, "Simulated Annealing: A Proof of Convergence," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* Vol. 16, No. 6, pp. 652-656.

[57] vanLaarhoven, P. J. M., E. H. L. Aarts and J. k. Lenstra, 1992, "Job Shop Scheduling by Simulated Annealing," *Operations Research,* Vol. 40, No. 1, pp. 113-125.

[58] Cerny, V., 1985, "Thermodynamical Approach to the Traveling Salesman Problem: An Efficient Simulation Algorithm," *Journal of Optimization Theory and Applications,* Vol. 45, No. 1, pp. 41-51.

[59] Alfa, A., S. Heragu and M. Chen, 1991, "A 3-Opt Based Simulated Annealing Algorithm for Vehicle Routing Problems," *Computers and Industrial Engineering,* Vol. 21, pp. 635-639.

[60] Kampke, T. K., 1988, "Simulated Annealing: Use of a New Tool in Bin Packing," *Annals of Operations Research,* Vol. 16, No. 327-332.

[61] Athier, G., P. Floquet, L. Pibouleau and S. Domenech, 1996, "Optimization of Heat Exchanger Networks by Coupled Simulated Annealing and NLP Procedures," *Computers and Chemical Engineering,* Vol. 20, pp. S13-S18.

[62] Corana, A., M. Marchesi, C. Martini and S. Ridella, 1987, "Minimizing Multimodal Functions of Continuous Variables with the "Simulated Annealing" Algorithm," *ACM Transactions on Mathematical Software,* Vol. 13, No. 3, pp. 262-280.

[63] Holland, J. H., 1975, *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence,* University of Michigan Press, Ann Arbor.

[64] Goldberg, D. E., 1989, *Genetic Algorithms in Search, Optimization, and Machine Learning, 1st Ed.,* Addison-Wesley, MA.

[65] Chen, C. L., V. S. Vempati and N. Aljaber, 1995, "An Application of Genetic Algorithms for Flow Shop Problems," *European Journal of Operational Research,* Vol. 80, pp. 389-396.

[66] Kumar, K. S., R. Tiwari and P. V. V. N. Prasad, 2009, "An Optimum Design of Crowned Cylindrical Roller Bearings Using Genetic Algorithms," *Journal of Mechanical Design,* Vol. 131, pp. 051011-1-14.

[67] Alkhatib, R., G. N. Jazar and M. F. Golnaraghi, 2004, "Optimal design of passive linear suspension using genetic algorithm," *Journal of Sound and Vibration,* Vol. 275, pp. 665-691.

[68] Baumal, A. E., J. J. McPhee and P. H. Calamai, 1998, "Application of Genetic Algorithms to the Design Optimization of An Active Vehicle Suspension System," *Computer Methods in Applied Mechanics and Engineering,* Vol. 163, pp. 87-94.

[69] Glover, F., 1986, "Future Paths for Integer Programming and Links to Artificial Intelligence," *Computers and Operations Research,* Vol. 13, No. 5, pp. 533-539.

[70] Glover, F. and M. Laguna, 1997, *Tabu Search,* Kluwer, Norwell.

[71] Glover, F., 1989, "Tabu Search - Part I," *ORSA Journal on Computing,* Vol. 1, No. 3, pp. 190-206.

[72] Siarry, P. and G. Berthiau, 1997, "Fitting of Tabu Search To Optimize Functions of Continuous Variables," *Journal for Numerical Methods in Engineering,* Vol. 40, pp. 2449-2457.

[73] Tan, K. C., Y. H. Chew and L. H. Lee, 2006, "A Hybrid Multiobjective Evolutionary Algorithm for Solving Vehicle Routing Problem with Time Windows," *Computational Optimization and Applications,* Vol. 34, pp. 115-151.

[74] Korsvik, J. E. and K. Fagerholt, 2008, "A Tabu Search Heuristic for Ship Routing and Scheduling with Flexible Cargo Quantities," *Journal of Heuristics,* Vol. 16, No. 2, pp. 117-137.

[75] Abt, C., S. D. Bade, L. Birk and S. Harries, 2001, "Parametric Hull Form Design - A Step Towards One Week Ship Design," *8th International Symposium on Practical Design of Ships and Other Floating Structures*, Shanghai, China.

[76] Simpson, T. W., A. J. Booker, D. Ghosh, A. A. Giunta, P. N. Koch and R.-J. Yang, 2004, "Approximation Methods in Multidisciplinary Analysis and Optimization: A Panel Discussion," *Structural and Multidisciplinary Optimization,* Vol. 27, pp. 302-313.

[77] Simpson, T., D. K. J. Lin and W. Chen, 2002, "Sampling Strategies for Computer Experiments: Design and Analysis," *International Journal of Reliability and Application,* Vol. 2, No. 3, pp. 209-240.

[78] Hardy, R. L., 1971, "Multiquadric Equations of Topology and Other Irregular Surfaces," *Journal of Geophysical Research,* Vol. 76, No. 8, pp. 1905-1915.

[79] Parzen, E., 1962, "On Estimation of a Probability Density Function and Mode," *The Annals of Mathematical Statistics,* Vol. 33, No. 3, pp. 1065-1076.

[80] Scott, D. W., 1992, *Multivariate Density Estimation,* John Wiley & Sons, Inc., New York.

[81] Evans, A. G., J. W. Hutchinson, N. A. Fleck, M. F. Ashby and H. N. G. Wadley, 2001, "The Topological Design of Multifunctional Cellular Materials," *Progress in Materials Science,* Vol. 46, No. 3-4, pp. 309-327.

[82] Wadley, H. N. G., 2006, "Multifunctional Periodic Cellular Metals," *Philosophical Transactions of the Royal Society,* Vol. 364, No. 1838, pp. 31-68.

[83] Wang, Y. C. and R. S. Lakes, 2004, "Extreme Stiffness Systems due to Negative Stiffness Elements," *American Journal of Physics,* Vol. 72, No. 1, pp. 40-50.

[84] Huang, T. and A. S. Mohan, 2006, "Micro-particle swarm optimizer for solving high dimensional optimization problems," *Applied Mathematics and Computation,* Vol. 181, pp. 1148-1154.

[85] MacNish, C. and X. Yao, 2008, "Direction Matters in High Dimensional Optimization," *2008 IEEE Congress on Evolutionary Computation*, pp. 2377-2384.

[86] Parsopoulos, K. E., 2009, "Cooperative Micro-Differential Evolution for High-Dimensional Problems," *Proceedings of the 11th Annual Conference on Genetic and Evolutionary Computation*, Montreal Quebec, Canada.

[87] Koehler, J. R. and A. B. Owens, 1996, "Computer Experiments," *Handbook of Statistics* (S. Ghosh and C. R. Rao, Eds.), Elsevier Science, Amsterdam, Vol. 13**,** pp. 261-308.

[88] McKay, M. D., R. J. Beckman and W. J. Conover, 1979, "A Comparison of Three Methods for Selecting Values of Input Variables in the Analysis of Output from a Computer Code," *Technometrics,* Vol. 21, No. 2, pp. 239-245.

[89] Hammersley, J. M., 1960, "Monte Carlo Methods for Solving Multivariable Problems," *Annals of the New York Academy of Sciences,* Vol. 86, pp. 844-874.

[90] Owen, A. B., 1992, "Orthogonal Arrays for Computer Experiments, Integration, and Visualization," *Statistica Sinica,* Vol. 2, pp. 439-452.

[91] Fang, K. T., D. K. J. Lin, P. Winker and Y. Zhang, 2000, "Uniform Design: Theory and Applications," *Technometrics,* Vol. 42, No. 3, pp. 237-248.

[92] Shan, S. and G. G. Wang, 2010, "Metamodeling for High Dimensional Simulation-Based Design Problems," *Journal of Mechanical Design,* Vol. 132, No. 5.

[93] Shan, S. and G. G. Wang, 2010, "Turning Black-Box into White Functions," *ASME Design Engineering Technical Conferences*, Montreal, QC Canada, Paper Number DETC2010-28958.

[94] Kotsiantis, S. B., 2007, "Supervised Machine Learning: A Review of Classification Techniques," *Informatica,* Vol. 31, pp. 249-268.

[95] Murthy, S. K., 1998, "Automatic Construction of Decision Trees from Data: A Multi-Disciplinary Survey," *Data Mining and Knowledge Discovery,* Vol. 2, pp. 345-389.

[96] Furnkranz, J., 1999, "Separate-and-Conquer Rule Learning," *Artificial Intelligence Review,* Vol. 13, pp. 3-54.

[97] Zhang, G. P., 2000, "Neural Networks for Classification: A Survey," *IEEE Transactions on Systems, Man, and Cybernetics - Part C: Applications and Reviews,* Vol. 30, No. 4, pp. 451-462.

[98] Friedman, N., D. Geiger and M. Goldszmidt, 1997, "Bayesian Network Classifiers," *Machine Learning,* Vol. 29, pp. 131-163.

[99] Burges, C. J. C., 1998, "A Tutorial On Support Vector Machines for Pattern Recognitions," *Data Mining and Knowledge Discovery,* Vol. 2, pp. 121-167.

[100] Pearl, J., 1988, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference, 2nd Ed.,* Morgan Kauffman Publishers, Inc., San Francisco.

[101] Koller, D. and N. Friedman, 2009, *Probabilistic Graphical Models: Principles and Techniques, 1st Ed.,* MIT Press.

[102] Reklaitis, G. V., A. Ravindran and K. M. Ragsdell, 1983, *Engineering Optimization: Methods and Applications,* John Wiley and Sons, New York.

[103] Deb, K. and M. Goyal, 1997, "Optimizing Engineering Designs Using a Combined Genetic Search," *Proceedings of the Seventh International Conference on Genetic Algorithms*, (Morgan Kauffman Publishers, San Francisco), pp. 521-528.

[104] Larrañaga, P., C. M. H. Kuijpers, R. H. Murga, I. Inza and S. Dizdarevic, 1999, "Genetic Algorithms for the Travelling Salesman Problem: A Review of Representations and Operators," *Artificial Intelligence Review,* Vol. 13, pp. 129-170.

[105] Goldberg, D. E. and K. Deb, 1991, "A Comparative Analysis of Selection Schemes Used in Genetic Algorithms," *Foundations of Genetic Algorithms* (G. J. E. Rawlins, Ed.)**,** pp. 69-93.

[106] Davis, L., 1985, "Applying Adaptive Algorithms to Epistatic Domains," *Proceedings of the International Joint Conference on Artificial Intelligence*, 162-164.

[107] Morales, K. A. and C. C. Quezeda, 1998, "A Universal Eclectic Genetic Algorithms for Constrained Optimization," *Proceedings of the 6th European Congress on Intelligent Techniques & Soft Computing*, EUFIT'98, 518-522.

[108] Rish, I., 2001, "An Empirical Study of the Naive Bayes Classifier," *IJCAI 2001 Workshop on Empirical Methods in Artificial Intelligence*.

[109] Zhang, H., 2004, "The Optimality of Naive Bayes," *Proceeding of the Seventeenth International Florida Artificial Intelligence Research Society Conference*, AIAA Press, Maimi Beach.

[110] Frank, M. V. and D. Helmick, 2007, "21st Century HVAC System for Future Naval Surface Combatants - Concept Development Report," *Naval Surface Warfare Center - Carderock Division*, West Bethesda, MD, Machinery Engineering Directorate Technical Report NSWCCD-98-TR-2007/06.

[111] McFarland, J. and I. R. McNab, 2003, "A Long-Range Naval Railgun," *IEEE Transactions on Magnetics,* Vol. 39, No. 1, January 2003, pp. 289-294.

[112] McNab, I. R. and F. C. Beach, 2007, "Naval Railguns," *IEEE Transactions on Magnetics,* Vol. 43, No. 1, January 2007, pp. 463-468.

[113] Scott, M., 2003, "SAMPSON MFR Active Phased Array Antenna," *IEEE International Symposium on Phased Array Systems and Technology*, Boston, MA, pp. 119 - 123.

[114] Price, D. C., 2003, "A Review of Selected Thermal Management Solutions for Military Electronic Systems," *IEEE Transactions on Components and Packaging Technologies,* Vol. 26, No. 1, pp. 26-39.

[115] Cusanelli, D. S. and G. Karafiath, 2012, "Hydrodynamic Energy Saving Enhancements for DDG 51 Class Ships," *ASNE Day 2012*, Arlington, VA, February 9-12, 2012.

[116] Hebner, R. E., J. D. Herbst and A. L. Gattozzi, 2011, "Pulsed Power Loads Support and Efficiency Improvement on Navy Ships," *Naval Engineers Journal,* Vol. 122, No. 4, pp. 23-32.

[117] Rolls-Royce, 2002, "MT30 Fact Sheet," [Brochure] Ref: MP/46/Issue 6 Aug 02.

[118] Rolls-Royce, 2004, "RR4500 Fact Sheet," [Brochure] Ref: MMS/FS/47/08/2.

[119] York, 2012, "Model YMC² Magnetic Bearing Centrifugal Liquid Chillers Style A," [Brochure] Form 160.78-EG1 (912).

Vita

Peter Bond Backlund was born in Menlo Park, CA in 1983. After graduating from Menlo School in Atherton, CA, he went to the University of Arizona where he received his Bachelor of Science in Mechanical Engineering in 2006. Peter entered the mechanical engineering graduate program at the University of Texas at Austin in 2006, where he earned his Master of Science in 2008. Peter returned to the University of Texas at Austin in 2009 to pursue a Doctor of Philosophy in Mechanical Engineering. As a doctoral student, he was a member and officer of the Texas Triathlon team.

Permanent email address: pbacklund@utexas.edu

This dissertation was typed by the author.